НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
Кафедра автоматизованих систем обробки інформації та управління

УДК: 004

«До захисту допущено»
Завідувач кафедри
_____ *О.А.Павлов*
(підпис) (ініціали, прізвище)
“___” _____2019 р.

Дипломний проект
на здобуття ступеня бакалавра

з напряму підготовки *6.050101 «Комп’ютерні науки»*

на тему: «*Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання*»

Виконав: студент *4* курсу, групи *ІС-51в*
Шпаков Віктор Андрійович _____
(прізвище, ім’я, по батькові) (підпис)

Керівник *доц., к.т.н., доц. Жданова Олена Григорівна* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з графічної документації *ст.викл. Халус Олена Андріївна* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент *доц., к.т.н., доц. Рєпнікова Н.Б.* _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2019 року

Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет (інститут) *інформатики та обчислювальної техніки*
(повна назва)

Кафедра *автоматизованих систем обробки інформації та управління*
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки *6.050101 «Комп’ютерні науки»*

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)
“___” _____2019 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ СТУДЕНТУ

Шпакову Віктору Андрійовичу
(прізвище, ім’я, по батькові)

1. Тема проекту «Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання»
керівник проекту Жданова Олена Григорівна, *к.т.н., доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджена наказом по університету від “*04*”*червня* 2019 р. №1478-с
2. Строк подання студентом проекту “*05*”*червня* 2019 року
3. Вихідні дані до проекту
Технічне завдання

4. Зміст розрахунково-пояснювальної записки
1. Загальні положення: основні визначення та терміни, опис предметного середовища, огляд ринку програмних продуктів, постановка задачі
2. Інформаційне забезпечення: вхідні дані, вихідні дані, опис структури бази даних
3. Математичне забезпечення: змістовна та математична постановки задачі, обґрунтування та опис методу розв’язання
4. Програмне та технічне забезпечення: засоби розробки, вимоги до технічного забезпечення, архітектура програмного забезпечення, побудова звітів
5. Технологічний розділ: керівництво користувача, методика випробувань програмного продукту

5. Перелік графічного матеріалу

1. Схема структурна діяльності

2. Схема структурна послідовності

3. Схема структурна компонентів програмного забезпечення

4. Схема структурна пакетів програмного забезпечення

5. Схема бази даних

6. Креслення вигляду екранних форм

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання *«15» лютого 2019 року*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломного Проекту	Строк виконання етапів проекту	Примітка
1.	*Вивчення рекомендованої літератури*	*15.02.2019*	
2.	*Аналіз існуючих методів розв'язання задачі*	*20.02.2019*	
3.	*Постановка та формалізація задачі*	*12.03.2019*	
4.	*Розробка інформаційного забезпечення*	*17.03.2019*	
5.	*Алгоритмізація задачі*	*22.03.2019*	
6.	*Обґрунтування використовуваних технічних засобів*	*30.03.2019*	
7.	*Розробка програмного забезпечення*	*20.04.2019*	
8.	*Налагодження програми*	*27.04.2019*	
9.	*Виконання графічних документів*	*04.05.2019*	
10.	*Оформлення пояснювальної записки*	*18.05.2019*	
11.	*Подання ДП на попередній захист*	*22.05.2019*	
12.	*Подання ДП на основний захист*	*05.06.2019*	
13.	*Подання ДП рецензенту*	*06.06.2019*	

Студент _____ Шпаков В.А.
(підпис)

Керівник проекту _____ Жданова О.Г.
(підпис)

№ з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
			Документи текстові		
1	А4	ДП ІС-4227.1478-с.ПЗ	Пояснювальна записка	71	
2	А4	ДП ІС-4227.1478-с.ТЗ	Технічне завдання	11	
3	А4	ДП ІС-4227.1478-с.ПМВ	Програма та методика випробувань	13	
			Матеріали графічні		
4	А3	ДП ІС-4227.1478-с.ССД	Схема структурна діяльності	1	
5	А3	ДП ІС-4227.1478-с.ССП	Схема структурна послідовності	1	
6	А3	ДП ІС-4227.1478-с.ССК	Схема структурна компонентів	1	
			програмного забезпечення		
7	А3	ДП ІС-4227.1478-с.ССП	Схема структурна пакетів програмного	1	
			забезпечення		
8	А4	ДП ІС-4227.1478-с.СБД	Схема бази даних	1	
9	А3	ДП ІС-4227.1478-с.КЕ	Креслення вигляду екранних форм	1	

Зм.	Арк.	ПІП	Підп.	Дата	ДП ІС-4227.1478-с.ВДП		
Розробн.		Шпаков В.А.			Відомість дипломного проекту	Літ.	Лист 1 / Листів 1
Керівн.		Жданова О.Г.					
Консульт.						КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в	
Н/контр.		Халус О.А.					
Зав. каф.		Павлов О.А.					

Пояснювальна записка
до дипломного проекту

на тему: Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання

Київ – 2019 року

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка дипломного проекту складається з п’яти розділів, містить 9 рисунків, 11 таблиць, 2 додатків, 17 джерел.

Дипломний проект присвячений розробці системи динамічної аутентифікації користувача з використанням лінгвістичного моделювання.

У розділі загальних положень були розглянуті процеси діяльності, предметне середовище та наявні аналоги програмного продукту.

У розділі інформаційного забезпечення були розглянуті вхідні та вихідні дані, описана база даних.

У розділі математичного забезпечення введено поняття та визначення Байєсових мереж, як засобу моделювання динаміки процесів довільної природи, визначено математичний апарат та процедуру формування логічного висновку в мережі, а також була запропонована методика застосування БМ для вирішення задачі аутентифікації.

У розділі з програмного забезпечення описані основні засоби розробки комплексу задач, структура пакетів, їх взаємодія у вигляді послідовності та основні компоненти програми. Наведені вимоги до технічного забезпечення.

У технологічному розділі описана інструкція користувача та проведене тестування системи.

БАЙЄСОВІ МЕРЕЖІ, АУТЕНТИФІКАЦІЯ, НАВЧАННЯ БАЙЄСОВОЇ МЕРЕЖІ, ДИНАМІЧНА БІОМЕТРИЧНА АУТЕНТИФІКАЦІЯ.

ABSTRACT

Structure and scope of work. The explanatory note of the diploma project consists of five sections, containing 9 figures, 11 tables, 2 annexes, and 17 sources.

The diploma project is devoted to the development of a dynamic user authentication system through the use of linguistic modeling.

In the basic concepts section operating processes, data domain and provided analogues were examined.

In the information support section input and output data were identified, database was described.

In the section of the mathematical support, term and definition of Bayesian network as a modeling method of arbitrary nature processes' dynamics were given, mathematical tool and procedure of formation logical network conclusion were defined, methods of Bayesian network usage for solving the authentication task was suggested.

The software section describes the main tools for developing a set of tasks, structure of toolsets and their interoperability in the form of sequence and major components of the program. The requirements for technical support were listed.

The technology section describes the user's manual and tests a set of tasks.

BAYESIAN NETWORK, AUTHENTICATION, BAYESIAN NETWORK LEARNING, DYNAMIC BIOMETRIC AUTHENTICATION.

ЗМІСТ

ВСТУП **5**

1 АНАЛІЗ СУЧАСНИХ РІШЕНЬ ПРОБЛЕМИ АУТЕНТИФІКАЦІЇ КОРИСТУВАЧА КОМПЬЮТЕРНИХ СИСТЕМ **7**

1.1 Опис предметного середовища 7

1.1.1 Опис процесу діяльності *10*

1.1.2 Опис функціональної моделі *13*

1.2 Огляд наявних аналогів 14

1.3 Постановка задачі 16

1.3.1 Призначення розробки *16*

1.3.2 Мета та задачі розробки *16*

Висновок до розділу 18

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ АУТЕНТИФІКАЦІЇ **19**

2.1 Вхідні дані 19

2.2 Вихідні дані 19

2.3 Опис структури бази даних 19

Висновок до розділу 21

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ АУТЕНТИФІКАЦІЇ **22**

3.1 Змістовна постановка задачі 22

3.2 Математична постановка задачі 22

3.3 Обґрунтування методу розв'язання 24

3.4 Опис методів розв'язання 37

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ **43**

4.1 Засоби розробки 43

4.2 Вимоги до технічного забезпечення 45

4.2.1 Загальні вимоги *45*

4.3 Архітектура програмного забезпечення 46

4.3.1 Діаграма послідовності *46*

4.3.2 Діаграма пакетів *47*

4.3.3 Діаграма компонентів *47*

4.3.4 Специфікація функцій *48*

Висновок до розділу 50

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ **51**

5.1 Керівництво користувача 51

5.2 Випробування програмного продукту 56

5.2.1 Мета випробувань *56*

5.2.2 Загальні положення *56*

5.2.3 Результати випробувань *57*

Висновок до розділу 60

ЗАГАЛЬНІ ВИСНОВКИ **61**

ПЕРЕЛІК ПОСИЛАНЬ **62**

ДОДАТОК А **64**

ВСТУП

Розвиток технологій, зокрема обчислюваної техніки привів до проникнення комп'ютерних систем у різноманітні галузі. Внаслідок цього, через локальні та глобальні мережі стали доступними цінні та важливі ресурси. Прикладом цього можуть бути: інтернет-банкінг, корпоративні программи, електронна комерція – вони містять цінну та закриту інформацію, або звичайний домашній комп'ютер з доступом до глобальних мереж може містити певну персональну інформацію. Тому проблема персоналізації користувача, що має доступ до певних ресурсів, стоїть дуже різко. Вирішення цієї проблеми за допомогою логіна і пароля є вразливим і не достатньо ефективним. Для розв'язання цієї проблеми запропоновано низку технологій, що ґрунтуються на біометричних даних користувачів.

Перші методи вирішення цієї проблеми мають своє походження з криміналістики, де була розв'язана низка подібних проблем, це зокрема, визначення людини за відбитками пальців, долонь та по обличчю. Проте через високу вартість супутніх пристроїв і через вразливість цих методів для зловмисників, вказані технології не набули широкого розповсюдження.

Більшість інструментів розпізнавання образів ґрунтується на двох технологіях: машинне навчання (machine learning) і візуалізація (візуальне подання інформації). Байєсівські мережі (БМ) якраз і поєднують у собі ці дві технології. Це досить молодий напрям розвитку науки що з'явився на стику двох наук (1) теорії ймовірностей та (2) теорії графів (розділ дискретної математики), а термін БМ з'явився в 1985 р. Ідея впровадження БМ полягає у представленні причинно-наслідкових зв'язків процесу у вигляді графа.

Аналіз існуючих методів розпізнавання показав, що БМ, у порівнянні з популярними моделями "чорних скриньок", надають більш зрозуміле пояснення своїх висновків, припускають логічну інтерпретацію і модифікацію структури відношень між змінними задачі, а також дозволяють в явній формі враховувати попередній апріорний досвід експертів. Завдяки

вдалому представленню у вигляді графів БМ дуже зручні для розв’язання прикладних задач користувачів. Вони ґрунтуються на фундаментальних положеннях і результатах теорії ймовірностей, які розроблялися на протязі декількох сотень років, що забезпечує їм успіх в розв’язанні практичних задач.

Дипломна робота присвячена дослідженню динамічних байєсових мереж, розробці методи побудови байєсових мереж та ймовірнісного висновку, а також їх практичного застосування до розв’язання задачі аутентифікації користувача.

1 АНАЛІЗ СУЧАСНИХ РІШЕНЬ ПРОБЛЕМИ АУТЕНТИФІКАЦІЇ КОРИСТУВАЧА КОМПЬЮТЕРНИХ СИСТЕМ

1.1 Опис предметного середовища

Бурхливий розвиток сучасних інформаційних систем і технологій ставить нові виклики в галузі інформаційної безпеки, зокрема розпізнавання користувачів. Тому дуже різко постала проблема персоналізації користувача, котрий має право доступу до певних ресурсів. Вирішення цієї проблеми за допомогою логіна і пароля виявилось вразливим і не достатньо ефективним. Для розв’язання цієї проблеми дуже багато уваги приділено розробці систем розпізнавання користувачів як із використанням спеціального біометричного обладнання, так і без нього.

Перші методи розв’язання цієї проблеми прийшли із криміналістики, де була вирішена низка подібних проблем, це зокрема, методи аутентифікації за відбитками пальців і по обличчю. Проте через високу вартість супутніх пристроїв і через вразливість цих методів для зловмисників, вказані технології не набули широкого розповсюдження. Введення логіну та паролю є методом статичної аутентифікації, тобто знаючи логін та пароль зайти може будь-яка людина, тому цей спосіб має значну вразливість. Використання біометрики для аутентифікації, тобто де характеристикою є якась фізична особливість користувача, значно ускладнює відтворення біометричних даних для зловмисників. Проте основний недолік біометричних систем – вона повинна мати високу чутливість, щоб підтвердити авторизованого користувача, але відкинути зловмисника зі схожими біометричними параметрами. Разом з тим суттєвим недоліком є велика ціна таких систем.

Також один із недоліків біометричних даних – їх відтворюваність. Наприклад генетичний код людини є майже абсолютно унікальним, проте міститься буквально в кожній клітині людини і може бути легко отриманий

зловмисниками. Відбиток пальця може бути знятий зловмисниками і відтворений для проникнення у систему.

Байєсові мережі (БМ) представляють собою графічні моделі подій і процесів на основі об'єднання деяких результатів теорії ймовірностей і теорії графів. Вони тісно пов'язані з діаграмами впливу, які можна використати для прийняття рішень. Незважаючи на свою назву, ці мережі не обов'язково мають на увазі тісний зв'язок з байєсовими методами. Назва пов'язана, насамперед, з байєсовим правилом ймовірнісного висновку [6].

БМ з’явилися на стику двох наук: теорії ймовірностей та теорії графів (розділ дискретної математики). Термін “байєсова мережа” був запропонований Джуді Перлом в 1985 році, з метою акцентування трьох аспектів: об’єктивного природи вхідних даних; отримання достовірної інформації при застосуванні теореми Байєса; ідея застосування причин та наслідків, запропонована в 1763 році в роботі Томаса Байєса.

Томас Байєс (1702–1761 роки) одним з перших зацікавився ймовірністю настання подій у майбутніх випробуваннях, ґрунтуючись на інформації про минулі випробування. Саме теорема Байєса пов’язує апріорні та апостеріорні ймовірності причин після спостереження за наслідками. В 1763 році в роботі “Опис методу розв’язання задачі в рамках доктрини випадкових подій” (“An essay towards solving a problem in the doctrine of chances”), опублікованій його товаришем Річардом Прайсом (Richard Price, 1723-1791) в п’ятдесят третьому номері лондонського журналу філософського королівського товариства , була вперше застосована ймовірність в індуктивному сенсі та встановлені основи ймовірнісного судження та висновку. Своє застосування теорема Байєса знайшла в теорії ймовірностей одразу після опублікування в 1763 р.

Ідея впровадження БМ полягає в представленні причинно-наслідкових зв’язків процесу у вигляді графа. Треба зауважити, що ідея представлення причинно-наслідкових зв’язків у вигляді графів розглядалася набагато

раніше. В 1921 р. біолог–генетик Сьюел Райт (Sewall Wright, 1889-1988 р.) запропонував пат-аналіз (path analysis) – статистичний метод, який дозволяє оцінити ступінь взаємовпливу змінних у причинно-наслідковій моделі. Для цього він об'єднав лінійні регресійні моделі та спрямовані графи, пізніше ця ідея була розвинута Гербертом Сімоном (Herbert Simon, 1916-2001) та Хьюбертом Балоком (Hubert Blalock).

До впровадження терміну "байєсова мережа" Джуді Перлом в 1985 році, БМ застосовувалися під назвою каузальних мереж (causal network), тобто мережі з причинно-наслідковими зв'язками. Байєсовими вони стали завдяки застосуванню в каузальних мережах теореми Байєса.

Застосування байєсових мереж для аналізу процесів різної природи, діяльності людини та функціонування технічних систем дозволяє враховувати та використовувати будь-які вхідні дані – експертні оцінки і статистичну інформацію. В свою чергу змінні можуть бути дискретними і неперервними, а характер їх надходження при аналізі та прийнятті рішення може бути як в режимі реального часу так і у вигляді статистичних масивів інформації і баз даних. При цьому, завдяки використанню представлення взаємодії між факторами процесу у вигляді причино-наслідкових зв'язків в мережі, досягаються максимально високий рівень візуалізації та, як наслідок, чітке розуміння суті взаємодії факторів процесу між собою. Іншими перевагами БМ є можливості врахування невизначеностей статистичного, структурного і параметричного характеру, а також формування висновку за допомогою різних методів – наближених і точних. Загалом, можна сказати, що БМ – це високоресурсний метод ймовірнісного моделювання процесів довільної природи з невизначеностями різних типів, який забезпечує можливість достатньо точного опису їх функціонування, оцінювати прогнози та будувати системи управління.

Відомі застосування БМ у системах технічної діагностики – система моніторингу космічного корабля багаторазового використання, діагностика

двигунів різних типів та призначення, аналіз стану технологічних процесів і технічних систем [3]. Широке застосування знаходять БМ в системах класифікації даних різної природи, системах автоматичного розпізнавання мовних сигналів, маркетингу і бізнесі, а також у багатьох інших сферах діяльності [4]. Загалом БМ дає можливість встановити причинно-наслідкові зв'язки між подіями та визначити ймовірності настання тієї чи іншої ситуації при отриманні нової інформації стосовно зміни стану будь-якого вузла (змінної) мережі. Ступінь успішності застосування даного методу моделювання та формування статистичного висновку залежить від вміння коректно сформулювати постановку задачі, вибрати змінні процесу, які в достатній мірі характеризують його динаміку або статику, зібрати статистичні дані та використати їх для навчання мережі, а також коректно сформувати результат – висновок за допомогою побудованої мережі. Побудова БМ пов'язана з необхідністю послідовного розв'язання декількох задач, зокрема це задачі обчислювального характеру, що зустрічаються при навчанні мережі. В загальному випадку навчання мережі відноситься до NP - повних задач, тобто об'єм обчислень зростає поліноміально із збільшенням кількості вузлів (змінних) мережі [4].

1.1.1 Опис процесу діяльності

Основною функціональністю системи, яка розробляється, є забезпечення можливості реєстрації та аутентифікації користувача, за допомогою Байєсових мереж та виведення результатів аутентифікації.

Після запуску програми в виводиться веб-сторінка програми, на якій можна подивитися, скільки користувачів зареєстровано в системі, та обрати аутентифікацію користувача або реєстрацію нового. Якщо користувач не є зареєстрованим в системі, то він обирає реєстрацію. Вводить свій логін та пароль за допомогою миші. В цей час система зчитує точки траєкторії руху

курсору миші по екрану. Після чого відбувається аналіз отриманих даних: траєкторія ділиться на точки, вираховується швидкість, прискорення, кутова швидкість, радіус кривизни та інші характеристики. Для того, щоб зібрані дані були коректними, користувач повинен декілька разів ввести той самий пароль, після чого, на основі вибірки буде вирахувано характеристики даного користувача, після чого вони будуть записані у базу даних.

При аутентифікації користувача виконується наступне: користувач переходить на відповідну сторінку, набирає свій логін та пароль. Якщо такий користувач є зареєстрованим, тоді завантажуються його траєкторії з бази даних, після чого, за допомогою байєсової мережі порівнюються характеристики траєкторії, набраної під час аутентифікації, зі збереженими у базу даних. На основі цього порівняння виводиться результат аутентифікації.

Типова послідовність дій для процесу проведення аутентифікації наведена в частині графічного матеріалу у вигляді структурної схеми діяльності.

Систему можливо вдосконалити, записуючи траєкторії руху миші кожної успішної аутентифікації користувача, та додаючи їх до вже існуючої вибірки, таким чином коректуючи характеристики даного користувача в системі. Перевага такого вдосконалення може бути у тому, що біометричні дані, які ми збираємо, можуть змінюватися, проте недоліком є ризик, що може зменшити точність системи, що погано позначиться на загальній безпеці системи.

Також можливо було б надати користувачу можливість використовувати різні пристрої вводу, наприклад, комп’ютерну мишу та сенсорну панель: записати декілька зразків вводу для одного користувача, порівнюючи потім траєкторію під час входу у систему зі всіма зразками. Тобто надати можливість в разі купівлі нового пристрою вводу додавати нові характеристики користувача з цього пристрою. Але таке нововведення також зашкоджує безпеці системи, оскільки збільшує кількість зразків, що робить

систему слабшою. Можливо реалізувати компроміс – щоб кількість пристроїв для користувача задавав адміністратор, тоді якщо необхідна зручність – можна збільшити кількість пристроїв, якщо важливіша безпека – залишити тільки один пристрій вводу.

1.1.2 Опис функціональної моделі

Із системою буде взаємодіяти один актор – користувач.

На рисунку 1.1 наведено схему структурних варіантів виконання.

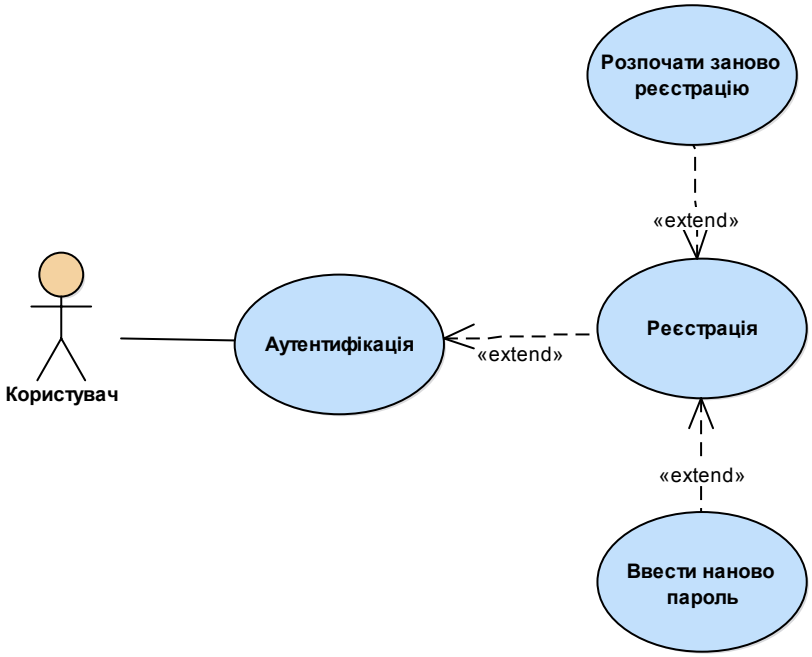


Рисунок 1.1 – Схема структурна варіантів використань

Таблиця 1.1 – Функції акторів

Актор	Функція	Опис функції
Користувач	1. Реєстрація	Користувач може зареєструватися у системі, для цього потрібно ввести логін та набрати пароль визначену системою кількість разів
	1.1. Аутентифікація	Користувач може бути аутентифікованим системою, ввівши логін та набравши пароль з екранної клавіатури
	1.1.1. Ввести наново пароль	Користувач може ввести пароль наново, якщо він помилився під час його набору
	1.1.2. Розпочати заново реєстрацію	Користувач може розпочати реєстрацію заново, з самого початку

1.2 Огляд наявних аналогів

Під час пошуку схожих за функціональністю систем було знайдено 2 програмних забезпечення, що вирішують задачу динамічної автентифікації користувача, яка використовує біометричні дані людей:

– KeyTrac;

– TypingDNA.

«KeyTrac» [17] - це універсальний, швидкий і легко інтегрований клавіатурний біометричний прикладний програмний інтерфейс, що створений для ідентифікації та аутентифікації користувача. Досліджує та аналізує динаміку натискання клавіш, не потребує ніякого спеціального обладнання.

KeyTrac Recorder пов’язаний з конкретними полями вводу і технічно не може записувати інформацію, яка знаходиться поза спеціальних текстових полів. Інша інформація, де б вона не з’явилася, буде повністю відкинута. KeyTrac записує виключно:

– відносний час натискання на клавішу;

– час переміщення від клавіші до клавіші;

– код клавіші.

Дані записуються за допомогою KeyTrack Recorder у фоновому режимі, отже і сам користувач не знає, що його можуть записувати.

KeyTrac може працювати практично з усіма веб-сторінками, оскільки використовує загальні галузеві стандарти, щоб забезпечити взаємодію з користувацькими системами.

Для ідентифікації користувача потрібно три кроки:

По-перше, KeyTrac збирає інформацію, як людина використовує свою клавіатуру. Це робиться за допомогою KeyTrac Recorder, який є невід'ємним компонентом, що використовується для ідентифікації динаміки натискань користувача.

Друге, після запису динаміки натискання клавіш необхідно передавати цей зразок набору в KeyTrac API, який потім обчислює оцінку відповідності на основі вже записаної вибірки набору, де йде порівняння з тими вибірками, що містяться в профілі користувача.

Третє, Після того, як обчислюється оцінка відповідності, KeyTrac API реагує на цю оцінку і рекомендує використовувати справжнє / хибне значення – відповідно, довіряти цій особі чи ні.

“TypingDNA API” [18] – це прикладний програмний інтерфейс, розроблений за технологією, яка розширює можливості обмеженої біометричної перевірки аутентичності, але не потребує спеціалізованих датчиків або дорогого обладнання. Він працює з існуючою клавіатурою, і може бути використаний для захисту будь-якого додатка шляхом співставлення короткої вибірки набору користувача.

TypingDNA теж записує «час натискання та час польоту» - час, коли користувач натискає на клавішу, час між натисканнями та коди клавіш

Шаблони наборів практично неможливо викрасти, і їх також простіше використовувати для веб-додатків, ніж інші біометричні системи: якщо ви хочете перевірити ідентифікацію особи за допомогою іншої біометричної системи, у браузері потрібно буде мати спеціальний дозвіл, щоб запитувати дані перевірки з апаратного забезпечення, тому це стає дуже складним.

TypingDNA надає комерційну аутентифікаційну версію API для розробників для інтеграції набору біометрії шляхом реалізації класу Javascript. Розробники можуть зв'язати TypingDNA API з будь-яким корпоративним або споживчим додатком, включаючи SaaS, веб-додатки, онлайн-навчання та постачальники платіжних послуг, щоб забезпечити двофакторну автентифікацію.

Використовуючи свій алгоритм штучного інтелекту для аналізу часу друку та часу польоту, TypingDNA дозволяє застосувати аутентифікацію на основі будь-якого тексту, надаючи своїм клієнтам гнучкість, наприклад, для

надання рекламного повідомлення як зразкового тексту для перевірки автентичності типізації користувача.

Можемо побачити, що ці програми мають схожі функції, виконують одну задачу, при цьому використовуючи біометричні дані користувача під час роботи з клавіатурою, але жодна з них не використовує біометричні дані, що надходять під час користування комп'ютерною мишею.

1.3 Постановка задачі

1.3.1 Призначення розробки

Призначенням розробки є високоякісне розпізнавання користувача комп'ютерної системи за статистичними характеристиками траєкторії руху курсора миші, вдосконалення аутентифікації користувача для посилення безпеки та створення захисту від зловмисників.

1.3.2 Мета та задачі розробки

Метою розробки є покращення захисту даних та забезпечення високої якості розпізнавання користувача комп'ютерної системи або мережі за статистичними характеристиками траєкторії руху курсора миші.

Для досягнення поставленої мети необхідно реалізувати наступні задачі:

- розробити метод оброблення траєкторних даних руху курсора з метою виділення описових статистик та інших ознак, необхідних для розпізнавання користувача;
- розробити модель для розпізнавання користувача у вигляді байєсової мережі (БМ) та методику її застосування до задачі розпізнавання;
- на основі запропонованих методів аналізу даних розробити інформаційну технологію для розпізнавання користувача ЕОМ;

– виконати експериментальні дослідження розробленої інформаційної технології.

Висновок до розділу

У даному розділі визначено головні завдання для дипломного проекту, наведено опис предметного середовища, дії, які можуть виконувати користувачі та функціональні можливості. Були продемонстровані аналогічні програми та їх особливості.

2 ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ АУТЕНТИФІКАЦІЇ

2.1 Вхідні дані

Вхідні дані надходять в застосування від користувача:

– логін користувача;

– пароль користувача;

– траєкторії рухів курсору миші.

Траєкторія зчитується наступним чином:

– зчитується висота точки краю курсору;

– зчитується ширина, на якій знаходиться точка краю курсору;

– та реєструється момент часу, коли відбулося зчитування координат точки краю курсору.

Всі інші дані вираховуються від вищенаведених. Логін та пароль перевіряються або вносяться в базу даних, з отриманих координат точок вираховується дистанція між ними, швидкість руху курсору, відхилення від прямої, кривизна його руху тощо.

2.2 Вихідні дані

Вихідними даними є результати роботи програми у вигляді повідомлень, що надходять користувачу:

– повідомлення про результат проходження аутентифікації користувача;

– повідомлення про помилки користувача.

При реєстрації введені дані: логін, пароль, та вирахувані характеристики траєкторії руху заносяться у базу даних.

2.3 Опис структури бази даних

У таблицях 2.1, 2.2 наведена структура таблиць бази даних.

Таблиця 2.1 – Опис таблиці “Користувачі”

Код	Опис	Тип даних	Обов'язкове	Унікальне	Первинний ключ	Зовнішній ключ
ID	Ідентифікатор користувача	INTEGER	X	X	X	
USERNAME	Ім’я користувача	VARCHAR(255)	X			
PASSWORD	Пароль користувача	VARCHAR(255)	X			

Таблиця 2.2 – Опис таблиці “Вибірка”

Код	Опис	Тип даних	Обов'язкове	Унікальне	Первинний ключ	Зовнішній ключ
ID	Ідентифікатор вибірки	INTEGER	X	X	X	
USER_ID	Ідентифікатор користувача	INTEGER	X			X
SAMPLE	Вибірка	BLOB	X			

Висновок до розділу

У даному розділі описані вхідні та вихідні дані системи. Описано структури таблиць бази даних з інформацією по кожній таблиці, описом полів таблиці. Для роботи системи динамічної аутентифікації користувача необхідно дві зв’язаних між собою таблиці.

Розроблена схема бази даних яка використана для збереження даних системи. Визначені первинні та зовнішні ключі бази даних системи.

3 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ АУТЕНТИФІКАЦІЇ

3.1 Змістовна постановка задачі

Необхідно за статистичними даними руху користувача комп'ютерною мишею під час набрання паролю побудувати байєсову мережу, яка, в подальшому, буде визначати персоналізацію користувача під час аутентифікації. Постає задача аналізу та фільтрації даних, що надійшли, визначення параметрів, що будуть записані, а також налаштування чутливості системи.

3.2 Математична постановка задачі

В рамках розв'язання проблеми обробки вхідних даних існує 2 задачі:

– обчислення навчальних множин для навчання БМ і побудова моделі даних користувача;
– обчислення характеристик для проведення розпізнавання.

Для побудови БМ і ДМБ необхідно сформувати навчальну множину, що містить історію реалізацій значень вершин мережі. Тобто, необхідно перейти від розгляду координат руху курсору до множини векторів, що представляють собою набори ознак, котрі обчислюються із початкових координат. В якості таких ознак в альтернативних дослідженнях використовувались статистичні характеристики траєкторій, такі як середня швидкість руху курсора, середнє прискорення курсора, довжина траєкторії, радіус кривизни траєкторії та ін. Використання таких ознак в дослідженнях виявилось занадто неефективним, тому постала проблема пошуку нової системи ознак, що б відрізнялась наступними якостями:

– інформативність – ознаки повинні містити інформацію про користувача, котрому належить траєкторія;
– унікальність – значення ознак мусять буди суттєво різнитися для різних користувачів;

– стійкість – ознаки повинні буди стійкими до похибок, що вносяться під час отримання координат, а також до методики отримання координат.

Для розв’язання першої задачі необхідно провести попередній збір траєкторних даних для кожного користувача. Опишемо послідовність дій:

1) Збір координатних даних для декількох користувачів.
2) Обчислення на їх основі характеристик.
3) Фільтрація характеристик
4) Пошук максимумів кожної характеристики по всіх користувачам. Збереження максимумів і нормалізація характеристик.
5) Групування характеристик і обчислення матриць коваріацій характеристик .
6) Застосування методу головних компонент для отриманих власних чисел. Відсіювання найменших неінформативних чисел. Збереження даних про найменші власні числа в БД.
7) Об’єднання решти власних чисел у вектори ознак.
8) Дискретизація отриманих векторів. Збереження дискретних проміжків в БД.
9) Об’єднання дискретних векторів в навчальні множини.

Процес підготовки даних до розпізнавання проходить за схожою схемою. Проте, в результаті на виході отримується вектори для аутентифікації. Опис процесу:

1) Збір координатних даних, достатньої кількості для обчислення коваріаційних матриць.
2) Обчислення характеристик на основі отриманих координат).
3) Фільтрація характеристик.
4) Нормалізація характеристик раніше отриманими значеннями (із БД).
5) Групування характеристик і обчислення коваріаційних матриць.

6) Відсіювання найменших власних чисел (за результатами методу головних компонентів, що збережені в БД).
7) Об’єднання решти власних чисел у вектор і його дискретизація згідно із раніше отриманими дискретними проміжками.

3.3 Обґрунтування методу розв’язання

Байєсові мережі (БМ) представляють собою графічні моделі подій і процесів на основі об'єднання деяких результатів теорії ймовірностей і теорії графів. Вони тісно пов'язані з діаграмами впливу, які можна використати для прийняття рішень. Незважаючи на свою назву, ці мережі не обов'язково мають на увазі тісний зв'язок з байєсовими методами. Назва пов'язана, насамперед, з байєсовим правилом ймовірнісного висновку. В літературі байєсові мережі (BN – Bayesian networks) іноді зустрічаються під назвами байєсові мережі довіри (BBN – Bayesian belief networks) причинні мережі (causal networks) або ймовірнісні мережі (probabilistic networks). БМ представляють собою зручний інструмент для опису досить складних процесів і подій з невизначеностями. Вони виявилися особливо корисними при розробці та аналізі машинних алгоритмів навчання. Основною ідеєю побудови графічної моделі є поняття модульності, тобто розкладання складної системи на прості елементи. Для об'єднання окремих елементів у систему використаються результати теорії ймовірностей, які забезпечують моделі практичну дієздатність у цілому, а також дають можливість поєднувати графічні моделі з базами даних. Такий граф-теоретичний підхід до побудови моделі дає досліднику можливість будувати моделі процесів з множини сильно взаємодіючих змінних, а також створювати структури даних для наступної розробки ефективних алгоритмів їхньої обробки та прийняття рішень.

Незважаючи на те, що байєсовим мережам приділяється багато уваги в закордонній літературі, принципи їхньої побудови, навчання та використання

ще недостатньо освітлені у вітчизняних публікаціях, що істотно ускладнює їхнє розуміння й застосування.

Байєсова мережа (БМ) найкраще підходить для вирішення даної задачі, оскільки вона є інструментом моделювання та аналізу даних довільної природи.

БМ мають наступні переваги:

- можливість об’єднувати в одній моделі дискретні та неперервні змінні;
- можливість обробки великих масивів даних;
- врахування невизначеностей інформаційного та інших типів [5].

Нехай V – множина дискретних змінних $\{X_1,\ldots,X_n\}$, $n \geq 1$. Кожна змінна $X_i \in V$ може приймати значення з множини $\{x_{i1},\ldots,x_{ir_i}\}$, $r_i > 1, i = 1,\ldots,n$. Покладемо, що кожна змінна – це екземпляр V, і кожна множина змінних – це підмножина V, якщо не було зроблено спеціального застереження.

Ймовірнісна мережа (мережа Байєса) B на множині V – це пари $B = (B_S, B_P)$, де мережна структура B_S – це спрямований ациклічний граф (САГ) з вершинами для кожної змінної в V; B_P – множина таблиць умовних ймовірностей, зв'язаних із B_S. Для кожної змінної $X_i \in V$ множина B_P містить таблицю умовних ймовірностей $P(X_i|\pi_i)$, що перелічує імовірності для всіх значень X_i при усіх даних комбінаціях значень змінних у його батьківській множині π_i в мережевій структурі B_S; згодом, такі комбінації значень будуть називатися реалізацією. Мережа B представляє спільний розподіл ймовірностей $P(V)$, що визначається за допомогою ланцюгового правила для повної ймовірності:

$$P(V) = \prod_{i=1}^{n} P(X_i|\pi_i) \tag{3.1}$$

Таким чином, мережа Байєса складається з наступних понять та компонент:

- множина випадкових змінних і спрямованих зв’язків між змінними;

- кожна змінна може приймати одне значення із кінцевого набору взаємовиключних значень;
- змінні зі зв’язками створюють орієнтований граф без циклів;
- кожній змінній-нащадку A із змінними-батьками $B_1,...,B_n$ приписується таблиця умовних ймовірностей $P(A/B_1,...,B_n)$.

Якщо змінна A не має батьківських вершин на графі, то замість умовних ймовірностей (автоматично) автоматично використовуються безумовні ймовірності $P(A)$. Тобто:

$$P(A/\pi(A)) = P(A), \text{ якщо } \pi(A) = \emptyset \qquad (3.2)$$

Вимога до відсутності петель в графі є суттєвою – для графів з петлями в ланцюгах умовних ймовірностей в загальному випадку немає коректної схеми проведення обчислень - внаслідок нескінченної рекурсії.

На практиці нам необхідні розподіли тих змінних, що нас цікавлять, взятих окремо. Вони можуть бути отримані із співвідношення для повної ймовірності шляхом маргіналізації – сумування по реалізаціям усіх змінних, крім обраних.

З математичної точки зору мережа Байєса – це модель представлення існуючих та відсутніх ймовірнісних залежностей. При цьому, зв'язок $A \to B$ є причинним, коли подія A є причиною виникнення B, тобто, коли існує механізм, у відповідності з яким значення, що приймає A, впливає на значення, що приймає B. Найбільш розповсюдженою загальною задачею, що розв’язується за допомогою мереж Байєса, є логічний (ймовірнісний) висновок (вивід).

Відмітимо про схожість формули (3.1) із визначення мережі Байєса з формулою множення ймовірностей:

$$P(x_1,...,x_n) = \prod_{i=1}^{n} P(x_i / x_1,...,x_{i-1}) = P(x_1)P(x_2/x_1)...P(x_n/x_1,...x_{n-1}). \qquad (3.3)$$

Для доведення цього факту скористаємось наступним співвідношенням між сумісним і умовним розподілом:

$$P(A,B) = P(A/B)P(B) = P(B/A)P(A). \quad (3.4)$$

І справді, якщо продовжити (3.3) і замінити всі умовні ймовірності на обернені, отримаємо:

$$P(x_1)\frac{P(x_1,x_2)}{P(x_1)}\frac{P(x_1,x_2,x_3)}{P(x_1,x_2)}...\frac{P(x_1,...,x_n)}{P(x_1,...,x_{n-1})} = P(x_1,...,x_n).$$

Відмітимо, також, що для змінних A і C, незалежних за умови B, виконується співвідношення:

$$P(A/B) = P(A/B,C). \quad (3.5)$$

Тепер необхідно показати, що (3.3) не заперечує (3.1). Насправді навіть більше: враховуючи довільну послідовність змінних у (3.3), жорстку умову на ациклічність графа, та співвідношення (3.5), отримуємо, що (3.1) є частковим випадком (3.3).

Тепер перейдемо до опису апарату логічного висновку в мережах Байєса. Для цього, насамперед, введемо декілька понять і визначень.

Нехай в нас є мережа Байєса, визначена в розділі 2.2. Також, нехай встановлено всі необхідні умовні й апріорні ймовірності згідно її мережевої структури. Спершу, розділимо множину змінних V на три підмножини:

$X_i \in V_1$ - так звані цільові змінні, значення (ймовірнісний розподіл) яких нас цікавить;

$X_j \in V_2$ - змінні, значення яких відомі постійно;

$X_k \in V_3$ - змінні, значення яких на даний момент або невідомо (неможливо встановити), або ігноруються.

При цьому: $V_1 \cup V_2 \cup V_3 = V$, $V_i \cap V_j = \emptyset$, і множина V_1 має бути не порожньою.

Подібне розбиття множини змінних на три підмножини відповідає ідею логічного висновку, що буде подана згодом.

Із (3.4) отримуємо співвідношення, відоме як теорема Байєса:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}. \qquad (3.6)$$

Більш точніше, нехай у нас $A_1,\ldots,A_n$ - набір (повна група) несумісних взаємовиключних подій (або альтернативних гіпотез). Тоді апостеріорна ймовірність $P(A_j/B)$ кожної із подій A_j при умові, що відбулася подія B, виражається через апріорну ймовірність $P(A_j)$:

$$P(A_j/B) = \frac{P(B/A_j)P(A_j)}{P(B)} = \frac{P(B/A_j)P(A_j)}{\sum_{j=1}^{n} P(B/A_j)P(A_j)}, \qquad (3.7)$$

де $$P(B) = \sum_{j=1}^{n} P(B/A_j)P(A_j) = \sum_{j=1}^{n} P(A_j,B). \qquad (3.8)$$

Обернена ймовірність $P(B/A_j)$ називається правдоподібністю (likelihood), а знаменник $P(B)$ в формулі Байєса – свідоцтвом (evidence). Спільна ймовірність є найбільш повним статистичним описом спостережуваних даних. Сумісний розподіл представляється функцією багатьох змінних – по кількості змінних в задачі, що досліджуються. В загальному випадку цей опис потребує завдання ймовірності всіх можливих конфігурацій значень всіх змінних, що важко застосувати навіть у випадку кількох десятків мулевих змінних. В байєсових мережах, коли існує додаткова інформація про степені залежності або незалежності ознак, ця функція факторизується на функції меншого числа змінних згідно (3.1).

Тепер дамо визначення поняттю логічний висновок. Нехай сталася подія e, під якою будемо розуміти прийняття змінними з множини V_2 певних станів із власної множини станів: $\{X_{j_1} = x_{j_1 l_{j_1}},\ldots,X_{j_m l_{j_m}} = x_{j_m}\} \in V_2$. Отже, під логічним висновком далі будемо розуміти знаходження умовних ймовірностей наступного вигляду: $P(V_1/V_2,e)$, або більш детальніше: $P(X_{i_1} = x_{i_1 l_{i_1}},\ldots,X_{i_k} = x_{i_k l_{i_k}} \in V_1 / X_{j_1} = x_{j_1 l_{j_1}},\ldots,X_{j_m l_{j_m}} = x_{j_m} \in V_2)$, де індексами $\{i_1,\ldots,i_k\}$ пронумеровані змінні з V_1, $\{j_1,\ldots,j_m\}$ - змінні з V_2, а індекс l нумерує стан кожної

змінної з її множини станів.

Тобто, нас цікавлять ймовірнісні розподіли станів $X_{i_1} = x_{i_1 l_{i_1}}, ..., X_{i_k} = x_{i_k l_{i_k}}$ для змінних з V_1, за умови, що нам відомі стани всіх змінних $X_{j_1} = x_{j_1 l_{j_1}}, ..., X_{j_m l_{j_m}} = x_{j_m}$ з множини V_2 на даний момент часу. При цьому, ми ігноруємо значення змінних з множини V_3. Таким чином, процедура логічного висновку відповідає на наступне питання: які найбільш ймовірні стани змінних з V_1, якщо змінні з V_2 приймуть конкретні значення.

Зауважимо, що умовна ймовірність, подана у визначенні логічного висновку є обчислюваною величиною і має встановлюватись згідно відомих розподілів змінних мережі Байєса.

Механізм проведення логічного висновку засновано на формулі Байєса (3.6, 3.7), а також співвідношеннях (3.3) і (3.1):

$$P(V_1/V_2, e) = P(\{X_{i_1} = x_{i_1 l_{i_1}}, ..., X_{i_k} = x_{i_k l_{i_k}}\} \in V_1 / \{X_{j_1} = x_{j_1 l_{j_1}}, ..., X_{j_m l_{j_m}} = x_{j_m}\} \in V_2) =$$

$$=$$

$$P(X_{i_1}, ..., X_{i_k} \in V_1 / X_{j_1}, ..., X_{j_m} \in V_2) = \frac{P(X_{i_1}, ..., X_{i_k} \in V_1, X_{j_1}, ..., X_{j_m} \in V_2)}{P(X_{j_1}, ..., X_{j_m} \in V_2)} =$$

$$= \frac{\sum_{\forall X \in V_3} P(X_1, ..., X_n)}{\sum_{\forall X \in V_1 \cup V_3} P(X_1, ..., X_n)} = \frac{\sum_{\forall X \in V_3} \prod_{i=1}^{n} P(X_i / \pi_i)}{\sum_{\forall X \in V_1 \cup V_3} \prod_{i=1}^{n} P(X_i / \pi_i)}. \quad (3.9)$$

Зазначимо, що підсумовування (маргіналізація) в чисельнику і знаменнику проводиться по всім вершинам із вказаних множин і по усім можливим комбінаціям їх станів.

Можливі випадки, коли кількість змінних у множинах V_1 і V_3 достатньо велика і кожна змінна має велику кількість станів, то провести повну маргіналізацію по всім змінним і всім станам значно ускладнюється. Тобто, якщо множина V_3 містить n булевих змінних, то кількість можливих комбінацій їх станів складає 2^n, тобто зростає експоненційно із ростом

кількості змінних. Зрозуміло, що в певних випадках провести повну маргіналізацію неможливо, або ця процедура займе велику кількість машинного часу. Тому замість точного логічного висновку можливо скористатися наближеними методами оцінювання, наприклад, варіаційними методами чи різноманітними варіаціями методу Монте-Карло, зокрема методом вибірок із латинських гіперкубів.

А тепер дамо алгоритм для проведення точного логічного висновку, що використовується в нашій системі.

Нехай q_1 – це кількість усіх можливих реалізацій станів із множини змінних V_1, а q_3 – відповідно V_3. Таким чином, пронумеровано усі можливі комбінації станів змінних: $\{1,...,q_1\}$ – із множини V_1 і $\{1,...,q_3\}$ – множини V_3 відповідно. Усі стани змінних із множини V_2 нам відомі. Тоді:

1) Встановлюємо відповідні значення всіх змінних із V_2: $\{X_{j_1} = x_{j_1 l_{j_1}},..., X_{j_m l_{j_m}} = x_{j_m}\} \in V_2$. Індексом k будемо нумерувати комбінації станів змінних множини V_1, а індексом j - V_3. На початку $k = 1$;
2) Встановлюємо значення всіх змінних із V_1 згідно k-ї комбінації, а $j = 1$ і переходимо до обчислення деякої суми p_k;
3) Встановлюємо значення всіх змінних із V_3 згідно j-ї комбінації. Оскільки на даному кроці значення всіх змінних в мережі встановлено, то ми можемо обчислити спільну ймовірність згідно (3.1): $p_k = p_k + \prod_{i=1}^{n} P(X_i | \pi_i)$;
4) $j = j + 1$. Якщо $j > q_3$, то $k = k + 1$ і переходимо на крок 2, інакше повертаємось на крок 3. Якщо $k > q_1$, то виходимо із процедури.

Таким чином, спільна ймовірність k-ї комбінації змінних із V_1 оцінюється як $\frac{p_k}{\sum_{k=1}^{q_1} p_k}$, що і являє собою результат логічного висновку.

Також додамо, що нормування на повну суму всіх p_k є наслідком нормованості таблиці ймовірностей.

До цього моменту ми сприймали структуру Байєсової мережі і всі набори умовних ймовірностей як деяку даність згори. Проте, на практиці доводиться знаходити структуру мережі, яка найкращим чином описує певну предметну область. Звісно, можна долучити до цього процесу досвідчених експертів, що мають певні знання стосовно конкретної предметної області. Але, впровадити досвід експертів не завжди можливо, та й відстежувати усі зв'язки між змінними, в разі їх великої кількості також неможливо.

Тому ми переходимо до автоматизації процесу побудови байєсових мереж. Отже, нехай ми маємо певну базу знань D, що складається з набору багатовимірних векторів, які репрезентують інформацію про деяку предметну область:

$$D = \begin{pmatrix} x_1^{(1)} & \dots & x_1^{(N)} \\ x_2^{(1)} & \dots & x_2^{(N)} \\ . & . & . \\ x_n^{(1)} & & x_n^{(N)} \end{pmatrix}. \qquad (3.10)$$

При цьому, n - кількість змінних в системі, а N – кількість векторів. Тобто ми знаємо значення змінних в деякі моменти часу, а кожній змінній відповідає вершина в байєсовій мережі. Змінні приймають значення із певної множини значень: $X_i \in V$, $X_i = \{x_{i1},\dots,x_{ir_i}\}$, $r_i > 1, i = 1,\dots,n$. Те, що нам відомі значення змінних за деякий проміжок часу зовсім не означає, що ми їх можемо спостерігати в будь-який момент часу в майбутньому. Наприклад, якщо мережа Байєса описує певну медичну систему, а кожній вершині відповідає певний симптом, діагноз чи аналіз, то деякі їх значення можуть бути невідомими, хоча можливо поставити попередній діагноз за конкретними симптомами. В подальшому, лікар може уточнити діагноз, направивши пацієнта на необхідні аналізи. Однак, взаємозв'язок симптомів і результатів аналізів вже може на початковій стадії привести до певного діагнозу.

Тепер повернемося до побудови мереж. Під побудовою Байєсової мережі будемо розуміти знаходження мережевої структури, тобто встановлення усіх причинно-наслідкових зв'язків між змінними, і знаходження відповідних умовних ймовірностей $P(X_i|\pi_i)$ на множині D. Отже, процес побудови мережі зводиться до наступного:

- перш за все, необхідно проаналізувати предметну область і сформувати множину D;
- по-друге, необхідно визначитися із мірою, яка буде оцінювати конкретну мережеву структуру B_S на множині D – $L(B_S,D)$. Завдяки цій мірі із двох мережевих структур B_{S_1} і B_{S_2} ми можемо вибрати ту, яка найбільш краще описує множину D;
- далі, необхідно запропонувати ефективний механізм перебору усіх мережевих структур, для виявлення найкращої в термінах $L(B_S,D)$;
- і наостанок, потрібно вказати алгоритм обчислення усіх умовних ймовірностей $P_{D,B_S}(X_i|\pi_i)$ для кожної мережевої структури B_S на множині D.

Вибір міри оцінювання Байєсової мережі є надзвичайно важливим і суттєво залежить від поставленої задачі. Далі ми розглянемо найбільш популярну і часто використовуємо міру мінімальної довжини опису (Minimum Description Length, MDL) і її адаптацію до задачі розпізнавання образів.

Нехай D буде база даних екземплярів на V, і нехай N буде кількість екземплярів в D (3.10). Нехай B_S позначає мережеву структуру на V, і для кожної змінної X_i нехай π_i буде множина батьків X_i в B_S. Крім того, для кожної π_i нехай ϖ_{ij} позначає j-ту реалізацію π_i відносно D, $j=1,...,q_i, q_i \geq 0$. Нехай N_{ijk} буде кількість екземплярів у базі даних D, у якій змінна X_i має значення x_{ik}, і π_i реалізується як ϖ_{ij}. Нехай $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

Визначимо міру MDL:

$$L_{MDL}(B_S, D) = \frac{1}{2} K \cdot \log N - N \cdot H(B_S, D), \qquad (3.11)$$

де K - це кількість параметрів мережі. Тобто перший доданок обчислює кількість бітів, необхідних для зберігання мережі, а $\log N$ вказує кількість бітів для кожного параметра. Другий доданок вказує необхідну кількість бітів для повної репрезентації бази D мережевою структурою B_S. Мінімізуючи значення MDL, ми знаходимо оптимальне співвідношення між цими факторами.

Зрозуміло, що мережева структура B_S значним чином залежить від множини D. Тому, процес формування останньої, є дуже важливим етапом в побудові Байєсової мережі.

Отже, за допомогою міри MDL, задавши певний перебір можливих мережевих структур, можливо знайти ту, що архівую в себе всю інформацію із множини D, і при цьому займає значно менше пам'яті. Саме в цьому і полягає ідея навчання за допомогою міри MDL. Однак, не завжди такий підхід є прийнятним і дуже часто залежить від поставленої задачі, зокрема задачі розпізнавання образів.

Взагалі, якщо ми множину D представимо у вигляді n-вимірних векторів $\{\vec{x}^{(1)},...,\vec{x}^{(N)}\}$, то класичний варіант навчання складається із максимізації правдоподібності даних, як функції матричних елементів:

$$L = \frac{1}{n \cdot N} \sum_{i=1}^{n} \sum_{j=1}^{N} \log(P(X_i / \pi_i, \vec{x}^{(j)})), \qquad (3.12)$$

Ми можемо піти тим самим шляхом і у випадку задачі розпізнавання вимагати вже мінімізації помилки розпізнавання на навчаючих векторах $\{\vec{x}^{(1)},...,\vec{x}^{(N)}\}$. Для цього опишемо механізм розпізнавання за допомогою мережі Байєса. Отже, нехай $o_1,...,o_m$ - набір об'єктів, що класифікуються, тоді вершина X_1 відповідає за об'єкти, де кожному стану

цієї вершини відповідає один об'єкт. Вершини $\{X_2,...,X_n\}$ відповідають за множину ознак об'єктів, де кожна вершина відповідає за певну ознаку. Таким чином, множина D представляє собою ніщо інше як набір реалізацій певних ознак для певних об'єктів. Наприклад, перший вектор цієї множини $\vec{x}^{(1)} = \{x_1^{(1)},...,x_n^{(1)}\}$ вказує, що об'єкт $o_i = x_1^{(1)}$ може мати набір ознак $\{x_2^{(1)},...,x_n^{(1)}\}$. Нехай для кожного об'єкту o_i є N_i векторів спостережень.

Якщо ми знаємо певний вектор ознак $\vec{y} = \{y_2,...,y_n\}$, то ми можемо оцінити ймовірність його приналежності об'єкту під номером o_i: $\hat{P}(X_1 = o_i / X_2 = y_2,..., X_n = y_n)$. Останню ймовірність ми і будемо вважати результатом розпізнавання мережею Байєса вектора $\vec{y}$.

Загалом, якщо об'єктів $o_1,...,o_m$ декілька, тоді розвозок задачі класифікації для відомого вектора спостережень $\vec{y} = \{y_2,...,y_n\}$ виглядає наступним чином:

$$o_i = \arg\max_i \hat{P}(X_1 = o_i / X_2 = y_2,..., X_n = y_n) \tag{3.13}$$

У випадку задачі аутентифікації, для кожного об'єкта o_i будується своя БМ, а вершина X_1 кожної мережі приймає лише два стани: o_i і $\overline{o_i}$. Причому стан o_i означає приналежність вектора $\vec{y} = \{y_2,...,y_n\}$ об'єкту o_i, а стан - $\overline{o_i}$ навпаки, вектор, що спостерігається, вказаному об'єкту не належить. Сам розв'язок задачі аутентифікації об'єкта o_i зводиться до наступного порівняння:

$$\hat{P}(X_1 = o_i / X_2 = y_2,..., X_n = y_n) > \beta, \tag{3.14}$$

де β - певне порогове значення.

Повернемось до розгляду навчаючої вибірки D. В ідеальному випадку, розпізнавання кожного вектора з цієї вибірки має привести до наступного результату:

$$\forall j = 1,..., N:\ \hat{P}(X_1 = o_i / X_2 = x_2^{(j)},..., X_n = x_n^{(j)}) = \begin{cases} 1, & якщо \quad x_1^{(j)} = o_i \\ 0, & якщо \quad x_1^{(j)} \neq o_i \end{cases}, \tag{3.15}$$

Тобто, БМ повинна видавати ймовірність 1, якщо вектор спостережень справді належить даному об'єкту і 0, для решти об'єктів. Таким чином, похибка розпізнавання вектора $\vec{y}^{(k)} = \{y_2^{(k)},..., y_n^{(k)}\}$, $k = 1,....,N$, що належить об'єкту o_i становить:

$$\varepsilon^{(k)} = 1 - P(X_0 = o_i / X_1 = y_1^{(k)},..., X_n = y_n^{(k)}) \quad (3.16)$$

Саму ж задачу навчання можна перевизначити як задачу мінімізації похибки розпізнавання векторів $\vec{y}^{(k)} = \{y_2^{(k)},..., y_n^{(k)}\}$:

$$\sum_{k=1}^{N} \varepsilon^{(k)} \to \min \quad (3.17)$$

Подібний результат буде означати мінімальну похибку розпізнавання векторів множини D і той факт, що всі ознаки із навчаючих векторів повністю визначають і розрізняють всі об'єкти. А будь-яке відхилення від одиничного значення на навчаючому векторі буде призводити до збільшення похибки. З іншого боку, необхідно накласти функцію штрафу на мережеву структуру для недопущення появи вершин із великою кількістю батьків. Враховуючи бажання зменшити похибку розпізнавання на навчаючих векторах, а також досвід застосування міри MDL для навчання байєсових мереж, визначимо міру мінімальної похибки наступним чином:

$$L_{\mathrm{Re}c}(B_s, D) = \frac{1}{2} K(B_s) \cdot \log N - N \cdot H_{\mathrm{Re}c}(B_S, D), \quad (3.18)$$

де доданок $K(B_s)$, як і у випадку міри MDL, відповідає за штраф на розмір мережевої структури:

$$K(B_s) = \sum_{i=1}^{n} q_i \cdot (r_i - 1),$$

а $H_{\mathrm{Re}c}(B_S, D)$:

$$H_{\mathrm{Re}c}(B_S, D) = \sum_{j=1}^{N} (\frac{1}{2} + \frac{1}{2} \hat{P}(X_1 = x_1^{(j)} / X_2 = x_2^{(j)},..., X_n = x_n^{(j)})) \times$$

$$\times \log(\frac{1}{2} + \frac{1}{2} \hat{P}(X_1 = x_1^{(j)} / X_2 = x_2^{(j)},..., X_n = x_n^{(j)})) = \quad ,$$

$$= \sum_{j=1}^{N} (1 - \frac{1}{2} \varepsilon^{(j)}) \log(1 - \frac{1}{2} \varepsilon^{(j)})$$

Таким чином, чим точніше буде виконуватись співвідношення (3.17), тим ближче буде доданок в $H_{\mathrm{Re}c}(B_S, D)$ до нуля і тим краще байєсова мережа проводить розпізнавання навчаючих векторів, і, як наслідок, виконання (3.17). З іншого боку, має бути компроміс між якістю розпізнавання та розмірами мережі. Тим більше, що при використанні алгоритму евристичного пошуку виникає тенденція до збільшення батьківської множини саме однієї із вершин. Використання в мірі $L_{\mathrm{Re}c}(B_s, D)$ доданку $K(B_s)$ запобігає цьому недоліку створюючи більш розгалужену топологію мережі.

Але, у випадку аутентифікації є ускладнення, тому що окрім самої похибки розпізнавання існують ще два типи помилок розпізнавання, що мають різну вартість:

– помилка першого типу – коли реальний об’єкт не аутентифікується;
– помилка другого типу – коли сторонній об’єкт аутентифікується як реальний.

3.4 Опис методів розв'язання

Оброблюємо зібрані дані та вираховуємо наступні характеристики, що використовуються, наведено в табл. 3.1.

Таблиця 3.1 – Набір ознак, що використовуються

$Vx_i = \frac{x_{i+1} - x_{i-1}}{t_{i+1} - t_{i-1}}, i = 1 \div (n-1);$	Швидкість по *x*, або перша похідна *x*.
$Vy_i = \frac{y_{i+1} - y_{i-1}}{t_{i+1} - t_{i-1}}, i = 1 \div (n-1);$	Швидкість по *y*, або перша похідна *y*.
$Ax_i = \frac{Vx_{i+1} - Vx_{i-1}}{t_{i+1} - t_{i-1}}, i = 2 \div (n-2);$	Прискорення по *x*, або друга похідна *x*.
$Ay_i = \frac{Vy_{i+1} - Vy_{i-1}}{t_{i+1} - t_{i-1}}, i = 2 \div (n-2);$	Прискорення по *y*, або друга похідна *y*.
$d3x_i = \frac{ax_{i+1} - ax_{i-1}}{t_{i+1} - t_{i-1}}, i = 3 \div (n-3)$	Третя похідна *x*.
$d3y_i = \frac{ay_{i+1} - ay_{i-1}}{t_{i+1} - t_{i-1}}, i = 3 \div (n-3)$	Третя похідна *y*.
$K_i = \frac{Vx_i Ay_i - Vy_i Ax_i}{\sqrt{\left(Vx_i^2 + Vy_i^2\right)^3}}, i = 2 \div (n-2);$	Радіус кривизни траєкторії в точці *i*.
$Fi_i = K_i \lvert V_i \rvert, i = 2 \div (n-2);$	Кутова швидкість в точці *i*.
$dK_i = \frac{K_{i+1} - K_{i-1}}{t_{i+1} - t_{i-1}}, i = 3 \div (n-3);$	Перша похідна радіуса кривизни.
$dFi_i = \frac{Fi_{i+1} - Fi_{i-1}}{t_{i+1} - t_{i-1}}, i = 3 \div (n-3);$	Перша похідна кутової швидкості.

Процес фільтрації відкидає усі значення характеристик, що не попадають в інтервал:

$$f'_{i,j,m,k} \in [\mu_{m,k} - \alpha \cdot \sigma_{m,k}; \mu_{m,k} + \alpha \cdot \sigma_{m,k}],$$

де $f'_{i,j,m,k}$ - відфільтровані характеристики, $\sigma_{m,k}$ - дисперсія k-тої характеристики, а $\mu_{m,k}$ - середнє значення k-тої характеристики, параметр α також є зовнішнім і показує яку частину розподілу потрібно відфільтрувати.

Нормалізація здійснюється по глобальному максимуму серед усіх можливих наборів характеристик і користувачів [3]:

$$\tilde{f}_{i,j,m,k} = \frac{f'_{i,j,m,k}}{\max\limits_{i,j,m} | f'_{i,j,m,k} |}, \forall k,$$

де $\tilde{f}_{i,j,m,k}$ - нормалізована значення характеристики (після фільтрації). Тобто, для кожної характеристики шукається глобальний максимум, які будуть зберігатися в базі даних як параметри системи і використовуватись при подальших нормалізаціях.

Далі застосовуємо метод головних компонент.

Нехай $F = \{F_1,...,F_P\}$ - характеристики. Тоді $\tilde{f}_{i,j,m,p}$ - i-те значення j-го сегменту m-го об'єкту характеристики $F_p(.)$. Якщо ми абстрагуємось від сегментів і заново пронумеруємо всі отримані значення характеристик, то отримаємо $\tilde{f}_{i,m,p}$, де $i = 1,...,N$ – загальне значення кількості p-их характеристик по всім сегментам для m-го об'єкту. Візьмемо N_x – певна кількість значень $\tilde{f}_{i,m,p}$. Визначимо вектор $\vec{f}_{i,m}$ – P-вимірний вектор значень $\tilde{f}_{i,m,p}$, а $\vec{\bar{f}}_m = \frac{1}{N_x}\sum_{i=1}^{N_x} \vec{f}_{i,m}$ - P-вимірний вектор середніх значень. Тоді

$$\Sigma_m = \frac{1}{N_x}\sum_{i=1}^{N_x}\left(\vec{f}_{i,m} - \vec{\bar{f}}_m\right)\left(\vec{f}_{i,m} - \vec{\bar{f}}_m\right)^T$$

P-вимірна коваріаційна матриця N_x перших значень ознак (вибіркова коваріаційна матриця), де $(.)^T$ - операція транспонування [1].

Вектор власних чисел матриці Σ_m позначимо як $\vec{\lambda}_m = \{\lambda_{1,m},...,\lambda_{P,m}\}$. Будемо вважати, що компоненти вектора $\vec{\lambda}_m$ вже впорядковані в порядку зменшення. Тоді, застосовуючи критерій інформативності, отримуємо P' головних компонент $\vec{\lambda}'_m = \{\lambda_{1,m},...,\lambda_{P',m}\}$, що в подальшому використовуються в якості ознак. Але, оскільки для навчання необхідно отримати декілька векторів вигляду $\vec{\lambda}'_m$, то загальна кількість значень характеристик N має в декілька разів перевищувати значення N_x, тобто:

$$\frac{N}{N_x} = I >> 1. \qquad (3.19)$$

Оскільки N_x - зовнішній параметр, значення якого обирається із загальних міркувань про інформативність ознак і, здебільшого, має експериментально встановлюватись, то процесу навчання, а перед цим аналізу і обробки даних, має передувати етап збору даних. Ми не будемо зараз зупинятись детально на цифрах, скажемо лише, що процес збору даних має проводитись таким чином, щоб даних(значень характеристик N) було достатньо, тобто виконувалось (3.19) [1]. В цьому випадку ми для кожного об'єкту m отримуємо набір матриць коваріацій $\Sigma_{i,m}$ і векторів власних чисел $\vec{\lambda}_{i,m} = \{\lambda_{1,i,m},...,\lambda_{P,i,m}\}$, де $i = 1,...,I$. Повторимо всі попередні викладки для даного випадку.

Вибіркові матриці коваріацій ознак m-го об'єкту:

$$\Sigma_{i,m} = \frac{1}{N_x} \sum_{j=(i-1)\cdot N_x}^{i\cdot N_x} \left(\vec{f}_{j,m} - \vec{\bar{f}}_{i,m}\right)\left(\vec{f}_{j,m} - \vec{\bar{f}}_{i,m}\right)^T, \qquad (3.20)$$

де $\vec{\bar{f}}_{i,m}$ - вибіркове середнє:

$$\vec{\bar{f}}_{i,m} = \frac{1}{N_x} \sum_{j=(i-1)\cdot N_x}^{i\cdot N_x} \vec{f}_{j,m} \qquad (3.21)$$

Таким чином, ми отримуємо набір коваріаційних матриць $\Sigma_{1,m},...,\Sigma_{I,m}$, яким відповідають набір векторів власних чисел $\Lambda_m = \{\vec{\lambda}_{1,m},...,\vec{\lambda}_{I,m}\}$. Оскільки, застосовуючи метод головних компонент, ми маємо отримати набір векторів власних чисел однакової довжини, тобто P' для всіх векторів має бути однаковою, то для цього необхідно модифікувати міру інформативності для декількох векторів:

$$I'_{P'}(\Lambda_m) = \frac{\sum_{j=1}^{I} \lambda_{1,j,m} + ... + \sum_{j=1}^{I} \lambda_{P',j,m}}{\sum_{j=1}^{I} \lambda_{1,j,m} + ... + \sum_{j=1}^{I} \lambda_{P,j,m}} . \qquad (3.22)$$

Як бачимо, міра інформативності (3.22) додатково ще й залежить від об'єкту m. Це означає, що для різних об'єктів довжина векторів P' після застосування методу головних компонент може бути різною, що додатково урізноманітнює простір ознак.

Процес дискретизації для всіх об'єктів має проходити однаково, а значить і дискретні проміжки для всіх об'єктів мають бути однаковими. Для цього об'єднаємо множини векторів ознак Λ_m в одну множину наступним чином: $\hat{\Lambda} = \{\hat{\lambda}_i\}$, $\hat{\lambda}_i = \{\lambda_{i,1,1},...,\lambda_{i,N_1,1},\lambda_{i,1,2},...,\lambda_{i,N_2,2},...,\lambda_{i,1,M},...,\lambda_{i,N_M,M}\} = \{\hat{\lambda}_{i,j}\}$ - агрегація всіх значень i-ї компоненти вектора ознак по всім об'єктам, де $N_1,...,N_M$ - кількості векторів ознак для кожного об'єкта відповідно.

Найпростішим було б знайти $\min_j \hat{\lambda}_{i,j}$, $\max_j \hat{\lambda}_{i,j}$ і поділити цей проміжок на R однакових частин: $[a_1^{(i)}, a_2^{(i)}], [a_2^{(i)}, a_3^{(i)}], ..., [a_R^{(i)}, a_{R+1}^{(i)}]$, де $a_1^{(i)} = \min_j \hat{\lambda}_{i,j}$ і $a_{R+1}^{(i)} = \max_j \hat{\lambda}_{i,j}$, а $a_{j+1}^{(i)} = a_j^{(i)} + \frac{a_{R+1}^{(i)} - a_1^{(i)}}{R}$ $\forall j > 1$. Таким чином, для всіх i, тобто для всіх компонент вектора ознак обчислюються власні

дискретні проміжки. Тоді дискретизованим станом деякого значення $\hat{\lambda}_{i,j}$ буде стан k, що задовольняє умові:

$$k: \begin{cases} \hat{\lambda}_{i,j} >= a_k^{(i)} \\ \hat{\lambda}_{i,j} < a_{k+1}^{(i)} \end{cases}$$

Проте, досліди показують, що проводити таким чином дискретизацію недоцільно – занадто низька якість розпізнавання.[7] Поясненням цього факту є нерівномірний розподіл значень ознак по дискретним проміжкам, що призводить до значного скупчення ознак в декількох проміжках і втрату інформативності в процесі дискретизації. Очевидно, що найбільша інформативність відповідає рівномірному розподілу значень $\hat{\lambda}_{i,j}$ по проміжках $[a_1^{(i)}, a_2^{(i)}], [a_2^{(i)}, a_3^{(i)}], ..., [a_R^{(i)}, a_{R+1}^{(i)}]$, що відповідає максимальній ентропії. Тобто, якщо w_r - частота попадання $\hat{\lambda}_{i,j}$ в проміжок $[a_{r-1}^{(i)}, a_r^{(i)}]$, тоді:

$$H = -\sum_{\forall r} \omega_r \log \omega_r \to \max$$

В такому разі, сам процес дискретизації (3.8.3) лишиться незмінним, а зміниться тільки принцип побудови дискретних проміжків. Нехай $\hat{\lambda}_{i,j}$ має L значень для кожного виміру i. Тоді нам необхідно L значень рівномірно розподілити в проміжках $[a_1^{(i)}, a_2^{(i)}], [a_2^{(i)}, a_3^{(i)}], ..., [a_R^{(i)}, a_{R+1}^{(i)}]$, де $a_1^{(i)} = \min_j \hat{\lambda}_{i,j}$ і $a_{R+1}^{(i)} = \max_j \hat{\lambda}_{i,j}$. Алгоритм знаходження невідомих проміжків достатньо простий:

1) Розглядаємо по циклу всі значення $\hat{\lambda}_{i,j}$. В середньому на кожен проміжок повинно приходитись по L/R значень. Номер проміжку $k = 1$. Кількість точок в поточному проміжку $l = 0$, індекс $j = 0$;
2) Якщо $l = L/R$, то

   $a_{k=1}^{(i)} = \hat{\lambda}_{i,j}$, $k = k+1$, $j = j+1$, $l = 0$

   інакше $j = j+1$;

3) Якщо $j = L$, то вихід із алгоритму, інакше перехід на крок 2.

Таким чином ми розглянули процес дискретизації ознак, що є кінцевим етапом обробки даних. Відмітимо, що дискретні проміжки $[a_1^{(i)}, a_2^{(i)}], [a_2^{(i)}, a_3^{(i)}], ..., [a_R^{(i)}, a_{R+1}^{(i)}]$ – це збережувані значення, які будуть використовуватись для подальшої дискретизації векторів в процесі розпізнавання.

Висновок до розділу

У розділі математичного забезпечення наведено математичне обґрунтування методів, що застосовуються в даному комплексі задач. Сформульована змістовна та математична постановки задачі. Наведені алгоритм створення, навчання Байєсових мереж та аналізу отриманих характеристик.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Реалізація серверної частини веб-додатку була виконана за допомогою мови програмування Python з використанням технології Flask.

Python [9] – це потужна мова програмування, якою легко оволодіти. Вона має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис Пайтона, динамічна обробка типів, а також те, що це інтерпретована мова, роблять його ідеальним для написання скриптів та швидкої розробки прикладних програм у багатьох галузях на більшості платформ.

Основними архітектурними характеристиками мови є:

- повна інтроспекція;
- автоматичне керування пам'яттю;
- система обробки виключень;
- підтримка багато поточності;
- гнучкі структури даних.

Програма написана даною мовою буде виконуватися на будь-якій операційній системі однаково, адже мова Python - кросплатформна. Можуть інколи виникати невеликі відмінності, але їх можна легко передбачити, якщо ознайомитись з документацією Python на офіційному сайті.

Перевагою мови є наявність великого числа програмних модулів, що забезпечують різні додаткові можливості.

Flask [10] – фреймворк, що створенний для роботи з веб-додатками. Він написаний на скриптовій мові програмування Python. Відноситься до категорії «мікрофреймворків» - мінімалістичних каркасів веб-додатків, які надають лише базові функції.

В якості СКБД була використана SQLite та такі інструменти роботи з базою даних, як peewee.

SQLite[11] - полегшена реляційна система керування базами даних. Втілена у вигляді бібліотеки, де реалізовано багато зі стандарту SQL-92. Сирцевий код SQLite є загальнодоступним поширюється як суспільне надбання , тобто може використовуватися без обмежень та безоплатно з будь-якою метою, комерційною чи приватною. SQLite - це найпоширеніша в світі СКБД з великою кількістю додатків, включаючи декілька високопрофесійних проектів. На відміну від більшості інших баз даних SQL, SQLite не має окремих серверних процесів. SQLite читає та пише безпосередньо до звичайних дискових файлів. Формат файлу бази даних є крос-платформою - ви можете вільно копіювати базу даних між 32-бітними та 64-бітними системами або між різними операційними системами. Файли бази даних SQLite є рекомендованим форматом зберігання бібліотеки Конгресу США.

Для реалізації графічного користувацького інтерфейсу були використані HTML, CSS, JavaScript, Bootstrap, JSON.

HTML [12] – стандартна мова розмітки яка застосовується для веб-сторінок в мережі Інтернет. Переважна кількість сторінок написані на даній мові. На сьогоднішній день він зазвичай використовується с JavaScript та CSS. HTML – простий для розуміння та в використанні. Зазвичай, більшість браузерів підтримують HTML, ніж будь-яка інша мова веб-програмування.

CSS [13] – це одна з основних мов мережі, яка стандартизована в різних браузерах відповідно до специфікації. Метою CSS є надання веб-розробникам стандартного способу визначення, застосування та управління наборами характеристик стилю.

JavaScript [14] призначений для створення сценаріїв для HTML-сторінок, яка інтегрується з кодом HTML-сторінки і інтерпретується веб-браузером. Значною перевагою в використанні даної мови є її об’єктно-

орієнтованість. Розробник має можливість працювати з великою кількістю об'єктів, а саме форми, фрейми, документи. Кожен об'єкт має свої властивості та методи.

Bootstrap [15] – це фреймворк на основі HTML і CSS, що включає різні шаблони оформлення для форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, використовуючи розширення написані на мові JavaScript.

JSON [16] – це універсальні структури даних. Майже всі сучасні мови програмування підтримують їх в будь-якій формі.

4.2 Вимоги до технічного забезпечення

4.2.1 Загальні вимоги

Для правильної роботи даної програми до складу технічних засобів повинні входити:

а) комп'ютер з такою конфігурацією:
   1) процесор з тактовою частотою не нижче 1 ГГц;
   2) достатній об'єм оперативної пам'яті (не менше 256 МБ);
   3) інші складові можуть мати будь-які параметри, тому що вони не значним чином впливають на роботу програми;

б) додатково має бути встановлене таке програмне забезпечення:
   1) операційна система Windows XP і вище/Linux/MacOS;
   2) Python 3.6;.

в) комп'ютерна периферія, до складу якої входить:
   1) монітор;
   2) мишка;
   3) клавіатура.

Клієнтам необхідно мати пристрій який має підключення до інтернету та може запустити один з таких браузерів:

– Google Chrome версії 40+;

– Safari версії 5.1+;
– Mozilla Firefox версії 17+;
– Microsoft Edge;
– Opera 40+;
– або їх альтернативи.

4.3 Архітектура програмного забезпечення

4.3.1 Діаграма послідовності

Робота програми відбувається в наступній послідовності - користувач обирає реєстрацію чи вхід у систему, вводить вхідні параметри. Схема структурної послідовності демонструє взаємодію об’єктів системи аутентифікації, упорядковану за часом. Схема структурної послідовності, що наведена в графічному матеріалі відображає такі задіяні об'єкти, як App, Process, DB.

App – описує методи інтерфейса системи та взаємодію з користувачем.

Process – описує методи системи, де виконується основна логіка застосування.

DB – об’єкт, що реагує на команди Process та зберігає всі дані зі згенерованими задачами та вхідними даними до них.

Для того щоб зареєструватися чи пройти аутентифікацію користувач переходить на сторінку з відповідною формою, вводить логін та набирає пароль за допомогою миші. Дані відправляються на Process, який аналізує отримані траєкторії, визначає їх характеристики, проводить фільтрацію, нормалізацію тощо. Після чого під час реєстрації дані записуються до DB, а під час входу в систему порівнюються з вже існуючими даними DB .

Схема структурна послідовності програмного забезпечення наведена в частині графічного матеріалу.

4.3.2 Діаграма пакетів

У програмному продукті методи і класи розділені на 3 пакета:

- com.app – містить методи для взаємодії з інтерфейсом;
- com.process – містить методи зчитування траєкторії, аналізу та фільтрації характеристик, навчання байєсової мережі та розпізнавання користувача;
- com.db – містить в собі класи, які взаємодіють з базою даних.

Схема структурна пакетів програмного забезпечення наведена в частині графічного матеріалу.

4.3.3 Діаграма компонентів

Програмний продукт має різні структурні компоненти та інтерфейси.

Компоненти програмного продукту зв'язані з іншими компонентами через інтерфейси. Зв'язок відбувається через наявні інтерфейси кожної компоненти. Наведемо опис наявних інтерфейсів.

UI – інтерфейс, що оброблює графічний інтерфейс , що взаємодіє з користувачtм. Даний інтерфейс допомагає веб-серверу обробляти дані, що надсилає користувач;

WSGI – зручний інтерфейс, що перетворює інформацію в бітову форму та навпаки для трансферу по протоколу TCP/IP, реалізує стандарт Python для взаємодії з сервером;

CommonDB – інтерфейс, що надає зручні методи для взаємодії програмного продукту з базою даних;

GetData – інтерфейс, що інтерпретує дані з бази даних в об’єкти.

Схема структурна компонентів програмного забезпечення представлення системи наведена в частині графічного матеріалу.

4.3.4 Специфікація функцій

Таблиця 4.1 – Методи пакету app

Метод	Опис методу
index	Генерує головну сторінку застосування
signup	Генерує сторінку реєстрації
learn	Обробляє дані реєстрації та відправляє їх у базу даних
login	Генерує сторінку входу
verify	Обробляє дані входу та проводить їх через байєсову мережу для співставлення з навчальними вибірками

Таблиця 4.2 – Методи пакету process

Метод	Опис методу	Параметри	Опис параметрів
process_initial	Повертає координати курсору та час	raw_trajectory	Масив необроблених траєкторій
compute_deriv	Повертає координати курсору та час	times, vec	Аргумент функції та значення функції
compute_radius	Повертає радіус кривизни траєкторії	speed_x, speed_y, accel_x, accel_y	Швидкість по *x*, швидкість по *y*, прискорення по *x*, прискорення по *y*
compute_angular_v	Повертає значення кутової швидкості	radius, speed_x, speed_y	Радіус кривизни траєкторії, швидкість по *x*, швидкість по *y*

Продовження таблиці 4.2

Метод	Опис методу	Параметри	Опис параметрів
filter	Повертає масив відфільрованих векторів параметрів	vecs, alpha	Масив векторів параметрів, число записаних траєкторій
get_best_discreet_intervals	Повертає масив інтервалів	vecs, interval_num	Масив векторів параметрів, кількість інтервалів
count_values	Повертає власні значення	vec, intervals	Вектор параметрів, масив інтервалів
process_sample	Повертає вектор параметрів руху	sample	Масив необроблених траєкторій
discreet	Визначає та повертає точки розбиття вибірки на інтервали	vecs, num_intervals	Масив векторів параметрів, кількість інтервалів
discreet_vec	Виконує дискретизацію вектора	vec, bounds, num_intervals	Вектор, масив меж інтервалів, кількість інтервалів
learn	Виконує навчання Байєсової мережі	user	Ім’я користувача
verify_user	Перевіряє користувача	user, sample	Ім’я користувача, вибірка траєкторій

Висновок до розділу

В даному розділі було наведено обґрунтування вибору засобів розробки. При написанні даної роботи було використано наступні технології: Python, JavaScript.

Описані вимоги до технічного забезпечення для коректної роботи програми.

Наведений опис структурної схеми пакетів та описана специфікація функцій що використовуються у всіх модулях.

Наведений опис структурної схеми послідовності, яка показує взаємодії об'єктів комплексу задач.

Описана схема структурна компонентів, що дозволяє визначити архітектуру розроблюваної системи, встановивши залежності між програмними компонентами.

5 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

5.1 Керівництво користувача

Для розуміння взаємодії користувача з системою продемонструємо процеси виконувані системою.

На рисунках 5.1-5.8 зображені скріншоти інтерфейсів системи динамічної аутентифікації.

Щоб запустити програмний застосунок необхідно:

– запустивши консоль, перейти у папку з програмою та набрати команду ‘flask run’;
– Набрати в адресному рядку браузера ‘http://localhost:5000’

При вході в систему відкривається стартова сторінка з можливими функціями:

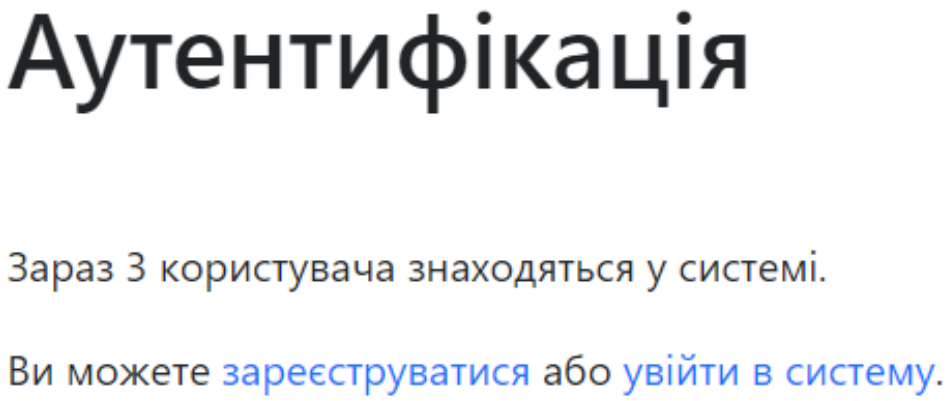


Рисунок 5.1 – Головне сторінка програмного застосування

Для того, щоб зареєструватися в системі, необхідно натиснути на гіперпосилання «Зареєструватися”. (Рисунок 5.2.)

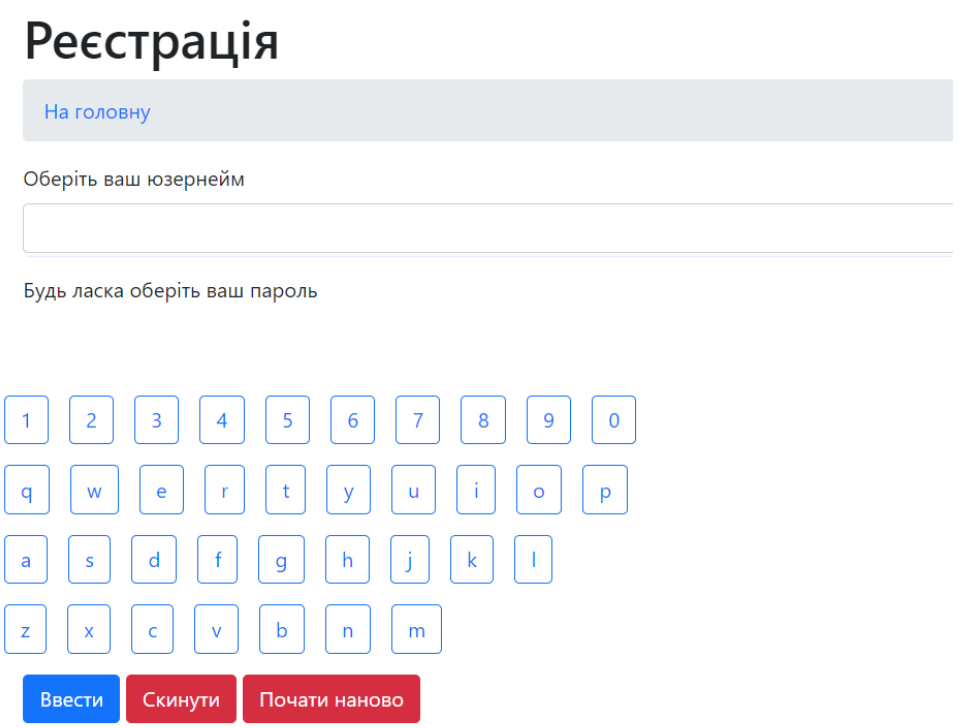


Рисунок 5.2 – Відкриття сторінки реєстрації

Впишемо логін та наберемо пароль за допомогою екранної клавіатури.

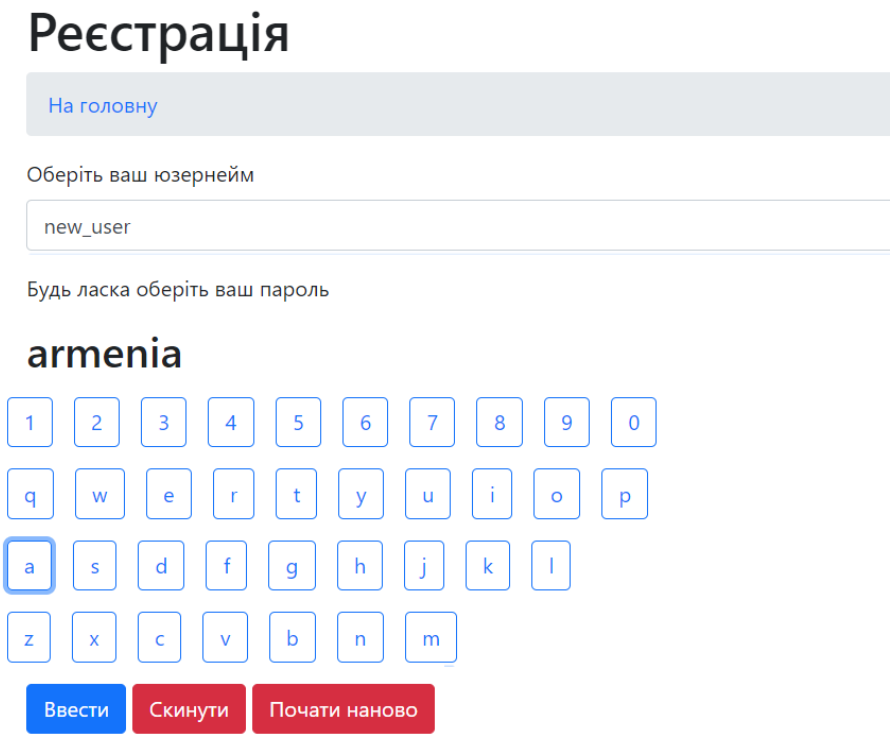


Рисунок 5.3 – Заповнена сторінка реєстрації

Натискаємо ввести та отримуємо повідомлення щодо підтвердження дії. (Рисунок 5.4.)

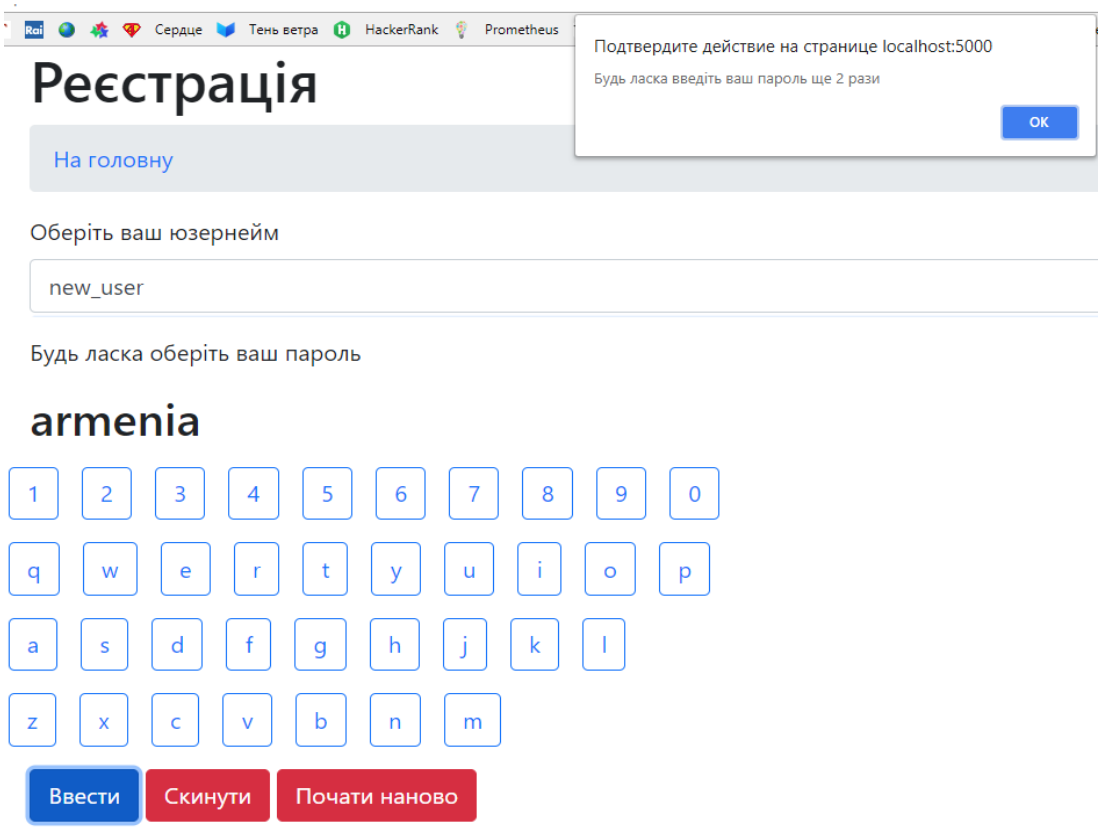


Рисунок 5.4 – Вікно реєстрації з повідомленням

Вводимо знову пароль визначену у повідомленні кількість разів, або вирішуємо припинити реєстрацію, для чого потрібно натиснути «Почати наново»: дані не збережуться, а користувач повернеться до головної сторінки. Після введення паролю потрібну кількість разів, програма повертає нас на головну сторінку (Рисунок 5.5.)

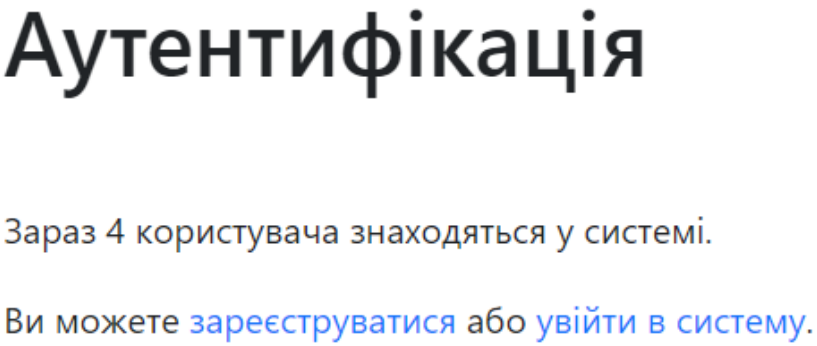


Рисунок 5.5 – Головна сторінка

Користувачів стало на один більше, тобто реєстрація була пройдена. Тепер пройдемо аутентифікацію. Для цього натиснемо на посилання «Увійти в систему». (Рисунок 5.6)

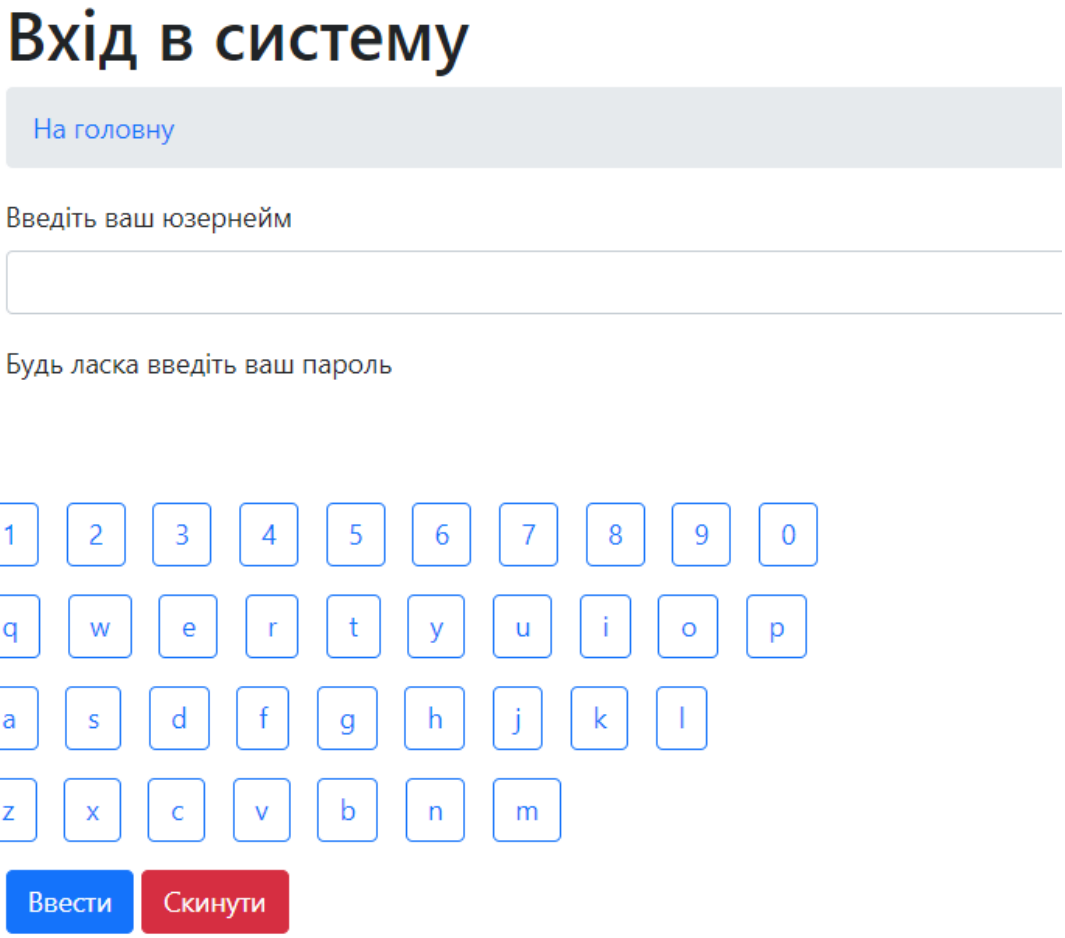


Рисунок 5.6 - Сторінка для аутентифікації

Заповнюємо поле логіну, після чого набираємо пароль на екранній клавіатурі (Рисунок 5.7)

Якщо під час набору пароля користувач допустив помилку, то може набрати пароль заново, натиснувши перед цим на кнопку «Скинути»

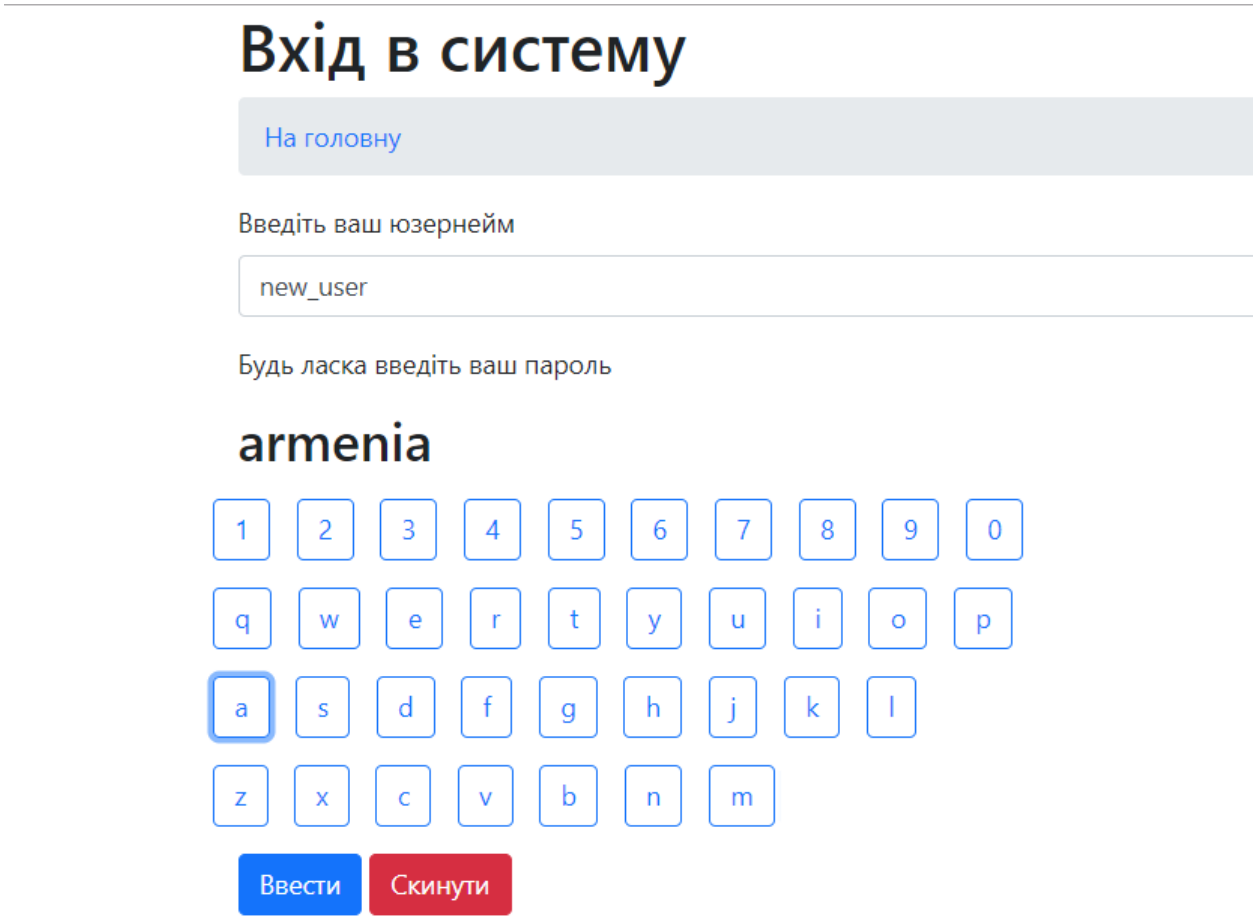


Рисунок 5.7 – Заповнена сторінка для аутентифікації

Далі натискаємо на кнопку “Ввести”(Рис. 5.8)

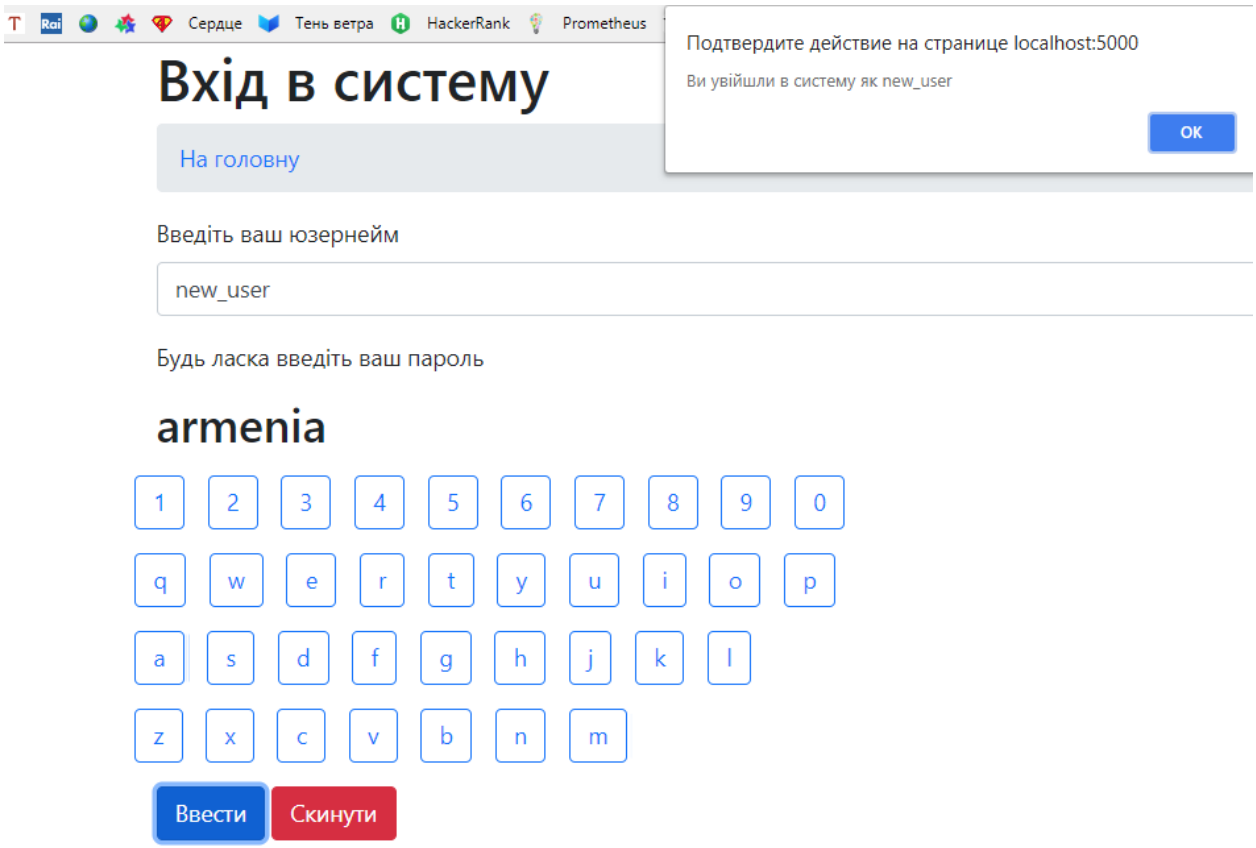


Рисунок 5.8 - Результати прохоження аутентифікації

Як бачимо з повідомлення – аутентифікація пройшла успішно.

5.2 Випробування програмного продукту

5.2.1 Мета випробувань

Метою випробувань являється перевірка відповідності функцій системи динамічної аутентифікації вимогам технічного завдання.

5.2.2 Загальні положення

Випробування проводяться на основі наступних документів:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;

– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5.2.3 Результати випробувань

В процесі тестування була перевірена уся функціональність сервісу. У наступних таблицях наведений перелік випробувань основних функціональних можливостей (табл. 5.1 – 5.5).

Таблиця 5.1 – Реєстрація користувача

Мета тесту:	**Перевірка функції «Реєстрація»**
Початковий стан КЗ	Відкрита головна сторінка
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Скільки раз набираємо пароль та натискаємо кнопку «Ввести»
Очікуваний результат:	Користувач зареєструвався
Стан КЗ після проведення випробувань:	Відкрита головна сторінка

Таблиця 5.2 – Відміна реєстрації

Мета тесту:	**Перевірка функції «Відміна реєстрації»**
Початковий стан КЗ	Відкрите вікно створення підсистеми
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Після цього натискаємо на кнопку “Почати наново”
Очікуваний результат:	Реєстрація не відбулась
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації

Таблиця 5.3 – Перевірка паролю в ході реєстрації

Мета тесту:	**Перевірка функції «Перевірка паролю в ході реєстрації»**
Початковий стан КЗ	Відкрите вікно створення підсистеми
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Після цього вводимо один раз неправильний пароль та натискаємо «Ввести»
Очікуваний результат:	Реєстрація не відбулась, Отримано повідомлення про помилку.
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації, з заповненим логіном та один раз введеним паролем

Таблиця 5.4 – Аутентифікація користувача

Мета тесту:	**Перевірка функції «Аутентифікація користувача»**
Початковий стан КЗ	Відкрита головна сторінка
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.
Схема проведення тесту:	Натиснути на посилання “Увійти в систему”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести»
Очікуваний результат:	Отримане повідомлення, що користувач увійшов в систему
Стан КЗ після проведення випробувань:	Відкрита головна сторінка

Таблиця 5.5 – Перевірка паролю під час аутентифікації

Мета тесту:	**Перевірка функції «Перевірка паролю під час аутентифікації»**
Початковий стан КЗ	Відкрита головна сторінка
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.
Схема проведення тесту:	Натиснути на посилання “Увійти в систему”, після цього ввести свій логін та набрати невірний пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести»
Очікуваний результат:	Отримане повідомлення, що доступ не надано, користувач не увійшов в систему
Стан КЗ після проведення випробувань:	Відкрита сторінка входу в систему

Висновок до розділу

В даному розділі були описані загальні інструкції користувача, проведено випробування продукту та наведено детальний опис кожного результату випробування. Була описана поведінка системи при введені коректних і некоректних даних. Беручи до уваги результати випробування, можна стверджувати, що система працює коректно та відповідає вимогам до програмного забезпечення.

ЗАГАЛЬНІ ВИСНОВКИ

У ході виконання дипломного проекту були детально розглянуті питання динамічної аутентифікації користувача за допомогою Байєсової мережі. Були визначені головні завдання дипломного проекту, наведено опис діяльності та функціональні можливості.

Розділ загальних положень надає інформацію про предметне середовище роботи. Описані діючі актори системи та їх функції. Проаналізовані аналоги програмного продукту. Постає дві проблеми: створення і навчання мережі Байєса та розпізнавання образів за допомогою Байєсової мережі.

Розділ математичного забезпечення присвячений формулюванню змістовної та математичної постановки задачі створення Байєсової мережі. Наведено опис алгоритмів розв’язання задачі розпізнавання. Розроблено методику застосування мереж Байєса до розв’язання задачі розпізнавання образів при розпізнаванні користувача комп’ютера в реальному часі. Мережа використовує статистичні ознаки для розпізнавання користувача, оцінені на основі траєкторних даних. Також були наведені приклади для подальшого вдосконалення системи, як додавання траєкторних даних успішної аутентифікації до вибірки та додавання інших пристроїв вводу.

Для розробки програмного забезпечення була використана мова Python програмної платформи PyCharm.

У розділі програмної та технічної підтримки описані використані в продукті технології. Розроблена модель бази даних для вирішення поставлених задач, для управління базою даних була обрана СУБД SQLite. Наведені мінімальні вимоги до технічного забезпечення. Розглядаються описи пакетів на яких базується архітектура розробки.

Наведена детальна інструкція користувача по експлуатації комплексу задач, описана методика проведення випробувань, яка показує можливість введення програми в експлуатацію.

ПЕРЕЛІК ПОСИЛАНЬ

1. Бідюк П. І. Застосування Байєсових мереж до розв'язання задачі класифікації / Бідюк П. І., Баклан Я. І. // «Системні технології», №3 (68) від 2010 р., Дніпропетровськ, 2010 р., 210 с. – С. 68-73.
2. Бідюк П. І. Застосування Байєсових і динамічних Байєсових мереж до розв'язання задачі класифікації і аутентифікації об'єктів / Бідюк П. І., Баклан Я. І. // Збірник наукових праць «Проблеми інформаційних технологій» №2 (006), 2009 р., . – С. 20-32.
3. Бідюк П. І. Алгоритми класифікації на основі Байєсових мереж / Бідюк П. І., Петренко В. В., Баклан Я. І. // Моделювання та прогнозування нелінійних динамічних процесів, Київ, «Екмо», 2004. 120 с. – С. 86 – 112.
4. Тулупьев А. Л. Байесовские сети: логико-вероятностный подход / А. Л. Тулупьев, С. И. Николенко, А. В. Сироткин – СПб.: Наука, 2006. – 607 с.
5. Городецкий В.И. Байесовский вывод / В.И. Городецкий. – Л.: ЛИИАН, 1991. – 149 с.
6. Тулупьев А. Л. Байесовские сети: вероятностная семантика и оптимизационные алгоритмы в логико-вероятностном выводе / А. Л. Тулупьев, Д. А. Никитин, С. И. Николенко / Материалы семинара "Информатика и компьютерные технологии", Санкт-Петербургский институт информатики и автоматизации Российской Академии наук. – 2004. – 74 с.
7. Тулупьев А. Л. Лекция 5: Введение в байесовские сети – логико-вероятностная модель баз фрагментов знаний с неопределенностью / А. Л. Тулупьев, А. В. Сироткин // Курс лекций по предмету "Алгоритмы для Интернета",

Математико-Механический факультет, Санкт-Петербургский государственный университет информационных технологий. – 2006. – 184 с.

8. Python [Електронний ресурс] // Режим доступу: https://docs.python.org/3/
9. Flask [Електронний ресурс] // Режим доступу: http://flask.pocoo.org/docs/0.12
10. SQLite [Електронний ресурс] // Режим доступу: https://www.sqlite.org/about.html
11. Advantages of HTML [Електронний ресурс] Режим доступу: https://www.flyingcowdesign.com/web-design-services/advantages-html
12. JavaScript [Електронний ресурс] Режим доступу: https://www.javascript.com/about
13. CSS Tutorial [Електронний ресурс] // Режим доступу: https://www.w3schools.com/Css/JavaScript [Електронний ресурс] // Режим доступу: https://documentation.js.org/
14. Bootstrap [Електронний ресурс] // Режим доступу: https://getbootstrap.com/docs/4.0/getting-started/contents/
15. JSON [Електронний ресурс] // Режим доступу: https://www.json.org/
16. KeyTrack [Електронний ресурс] // Режим доступу: https://www.keytrac.net/en/tour
17. TypingDNA[Електронний ресурс] // Режим доступу: https://www.typingdna.com/

Додаток А

Тексти програмного коду

Система динамічної аутентифікації користувача

(Найменування програми (документа))

DVD-R

(Вид носія даних)

8 арк, 4 291 Кб

(Обсяг програми (документа) , арк.,) Кб)

Київ – 2019 року

```
from flask import *

from db import *
from process import *

with database:
    database.create_tables([User, Sample])

app = Flask(__name__)


@app.before_request
def before_request():
    database.connect()


@app.after_request
def after_request(response):
    database.close()
    return response


qwerty_keys = ['1234567890', 'qwertyuiop', 'asdfghjkl', 'zxcvbnm']


@app.route('/')
def index():
    users = User.select()
    return render_template('index.html', keys=qwerty_keys, users=users)


@app.route('/sign-up')
def signup():
    return render_template('signup.html', keys=qwerty_keys)


@app.route('/learn', methods=['POST'])
def learn():
    data = json.loads(request.data)
    username = data['username']
    password = data['password']
    user = User.create(username=username, password=password)
    samples = data['samples']
    for sample in samples:
        data = process_sample(sample)
        Sample.create(user=user, sample=save_to_string(data))
    return jsonify({}), 200, {'Content-Type': 'application/json'}

@app.route('/login')
def login():
    return render_template('login.html', keys=qwerty_keys)


@app.route('/verify', methods=['POST'])
def verify():
    data = json.loads(request.data)
    try:
        user = User.get(User.username == data['username'])
    except DoesNotExist:
        return jsonify({'error': 'Invalid username or password'}), 400, {'Content-Type': 'application/json'}
    if user.password != data['password']:
        return jsonify({'error': 'Invalid username or password'}), 400, {'Content-Type': 'application/json'}
    data = process_sample(data['trajectory'])
```

```
 res = verify_user(user, data)
  print(res)
  return jsonify({'verification_status': [int(i[0]) for i in res]}), 200, {'Content-Type': 'application/json'}



from peewee import *


database = SqliteDatabase('db.sqlite3')

class User(Model):
  username = CharField()
  password = CharField()

  class Meta:
    database = database


class Sample(Model):
  user = ForeignKeyField(User)
  sample = BlobField()

  class Meta:
    database = database



import math
from io import BytesIO

import mdp

import numpy as np
from sklearn import svm

EPS = 0.0000001


def process_initial(raw_trajectory):
  t = np.array([i[0] for i in raw_trajectory])
  x = np.array([i[1] for i in raw_trajectory])
  y = np.array([i[2] for i in raw_trajectory])
  return t, x, y


def compute_deriv(times, vec):
  res = np.zeros(len(times))
  for i in range(1, len(vec) - 1):
    res[i] = (vec[i+1] - vec[i-1]) / (times[i+1] - times[i-1])
  return res


def compute_radius(speed_x, speed_y, accel_x, accel_y):
  res = np.zeros(len(speed_x))
  for i in range(len(speed_x)):
    denom = math.sqrt((speed_x[i] ** 2 + speed_y[i] ** 2) ** 3)
    if abs(denom) < EPS:
      continue
    res[i] = (speed_x[i] * accel_y[i] - speed_y[i] * accel_x[i]) / denom
  return res
```

```
def compute_angular_v(radius, speed_x, speed_y):
    res = np.zeros(len(radius))
    for i in range(len(radius)):
        res[i] = radius[i] * math.sqrt(speed_x[i] ** 2 + speed_y[i] ** 2)
    return res


def filter(vecs, alpha):
    means = []
    for i in vecs:
        means.append((np.mean(i), np.std(i)))
    bad_columns = set()
    for i in range(len(vecs[0])):
        for vec_i, vec in enumerate(vecs):
            mean, std = means[vec_i]
            if abs(vec[i] - mean) > alpha * std:
                bad_columns.add(i)
    rows = []
    for i in range(len(vecs[0])):
        row = []
        if i in bad_columns:
            continue
        for vec in vecs:
            row.append(vec[i])
        rows.append(row)
    return np.array(rows)


def get_best_discreet_intervals(vecs, interval_num):
    intervals = []
    cur_min = None
    cur_total = 0
    vec = np.concatenate(vecs)
    for val in sorted(vec):
        if cur_min is None:
            cur_min = val
        if cur_total >= len(vec) // interval_num:
            intervals.append((cur_min, val))
            cur_min = None
            cur_total = 0
        else:
            cur_total += 1
    return intervals


def count_values(vec, intervals):
    res = np.zeros(len(intervals), dtype='int')
    cur = 0
    for mi, ma in intervals:
        for value in vec:
            if mi <= value <= ma:
                res[cur] += 1
        cur += 1
    return res


def process_sample(sample):
    t, x, y = process_initial(sample)
    speed_x = compute_deriv(t, x)
    speed_y = compute_deriv(t, y)
    accel_x = compute_deriv(t, speed_x)
    accel_y = compute_deriv(t, speed_y)
    third_deriv_x = compute_deriv(t, speed_x)
    third_deriv_y = compute_deriv(t, speed_y)
```

```
    radius = compute_radius(speed_x, speed_y, accel_x, accel_y)
    angular_v = compute_angular_v(radius, speed_x, speed_y)
    deriv_radius = compute_deriv(t, radius)
    deriv_angular_v = compute_deriv(t, angular_v)

    vecs = [speed_x, speed_y, accel_x, accel_y, third_deriv_x, third_deriv_y, radius, angular_v, deriv_radius,
            deriv_angular_v]

    return vecs


def save_to_string(vec):
    io = BytesIO()
    np.save(io, vec)
    return io.getvalue()


def load_from_string(b):
    io = BytesIO(b)
    return np.load(io)


def discreet(vecs, num_intervals=100):
    mi = min([min(i) for i in vecs])
    ma = max([max(i) for i in vecs])
    step = (ma - mi) / num_intervals
    res = [np.zeros(num_intervals, dtype=int) for _ in vecs]
    for ix, vec in enumerate(vecs):
        cur = mi
        current_interval = 0
        for i in sorted(vec):
            if i > cur + step:
                current_interval += 1
                cur = cur + step
            if current_interval == num_intervals:
                continue
            res[ix][current_interval] += 1
    return mi, ma, res


def discreet_vec(vec, bounds, num_intervals=100):
    cur = bounds[0]
    current_interval = 0
    step = (bounds[1] - bounds[0]) / num_intervals
    res = np.zeros(num_intervals, dtype=int)
    for i in sorted(vec):
        if i > cur + step:
            current_interval += 1
            cur = cur + step
        if current_interval >= num_intervals:
            continue
        res[current_interval] += 1
    return res


def prepare_data(data):
    values = []
    for i, column in enumerate(data.T):
        values.append(discreet(column, 20)[1])
    res = list(np.array(values).T)
    return [['user'] * len(res[0])] + res


def learn(user):
```

```
    from db import Sample

sets = None

for sample in Sample.select().where(Sample.user == user):
    data = load_from_string(sample.sample)
    # 3 and 5 figured out experimentally, may not work well for all trajectories... Needs moar testing
    data = filter(data, 3)
    data = mdp.pca(data, output_dim=5)
    if sets is None:
      sets = []
      for i in data.T:
        sets.append([i])
    else:
      for i, col in enumerate(data.T):
        sets[i].append(col)

  models = []
  bounds = []
  for s in sets:
    mi, ma, data = discreet(s)
    bounds.append((mi, ma))
    m = svm.OneClassSVM(kernel="rbf", nu=0.1)
    m.fit(data)
    models.append(m)
  return bounds, models


def verify_user(user, sample):
  bounds, models = learn(user)
  data = filter(sample, 3)
  data = mdp.pca(data, output_dim=5)
  vecs = []
  for i, vec in enumerate(data.T):
    vecs.append(discreet_vec(vec, bounds[i]))
  res = []
  for i, model in enumerate(models):
    res.append(model.predict(vecs[i].reshape(1, -1)))
  return res


<script>
  var kb = $('#keyboard');
  var pos = kb.position();
  var result = '';
  var trajectory = [];
  var start_ts = null;
  var samples_needed = 10;
  var samples = [];
  var password = null;

  function update_result(key) {
    if (key !== undefined) {
      result += key;
    }
    if (result) {
      $('#result').text(result);
    } else {
      $('#result').html(' ');
    }
  }

  kb.on('mousemove', function (event) {
    if (start_ts) {
```

```
        var ts = new Date().getTime();
        trajectory.push([ts - start_ts, event.pageX - pos.left, event.pageY - pos.top]);
      }
    });
    $('#reset').on('click', function (event) {
      result = '';
      trajectory = [];
      update_result();
      start_ts = null;
    });
    $('#reset-all').on('click', function(event) {
      result = '';
      trajectory = [];
      update_result();
      start_ts = null;
      samples = [];
      password = null;
    });
    $('.key').on('click', function (event) {
      if (start_ts === null) {
        start_ts = new Date().getTime();
        trajectory = [];
      }
      update_result($(this).data('key'));
    });
    $('#finish').on('click', function (event) {
      start_ts = null;
      var username = $('#username').val();
      if (!username) {
        alert('Будь ласка введіть ваш юзернейм');
        result = '';
        update_result();
        return;
      }
      if (password === null) {
        password = result;
      } else if (result !== password) {
        alert("Паролі не співпадають. Зробіть усе з самого початку.");
        password = null;
        samples = [];
      }
      result = '';
      update_result();
      samples.push(trajectory);
      if (samples.length >= samples_needed) {
        $.ajax({
          method: 'POST',
          url: '{{ url_for('learn') }}',
          data: JSON.stringify({
            username: username,
            password: password,
            samples: samples
          }),
          contentType: 'application/json',
          success: function () {
            console.log('here');
            location.replace('{{ url_for('index') }}');
          }
        });
      } else {
        alert('Будь ласка введіть ваш пароль ще ' + (samples_needed - samples.length) + ' рази');
      }
    });
  </script>
```

```
<script>
  var kb = $('#keyboard');
  var pos = kb.position();
  var result = '';
  var trajectory = [];
  var start_ts = null;

  function update_result(key) {
    if (key !== undefined) {
      result += key;



  }
    if (result) {
      $('#result').text(result);
    } else {
      $('#result').html(' ');
    }
  }


  kb.on('mousemove', function (event) {
    if (start_ts) {
      var ts = new Date().getTime();
      trajectory.push([ts - start_ts, event.pageX - pos.left, event.pageY - pos.top]);
    }
  });
  $('#reset').on('click', function (event) {
    result = '';
    trajectory = [];
    update_result();
    start_ts = null;
  });
  $('.key').on('click', function (event) {
    if (start_ts === null) {
      start_ts = new Date().getTime();
    }
    update_result($(this).data('key'));
  });
  $('#finish').on('click', function (event) {
    start_ts = null;
    var username = $('#username').val();
    if (!username) {
      alert('Будь ласка введіть ваш юзернейм');
      result = '';
      update_result();
      return;
    }


    $.ajax({
      method: 'POST',
      url: '{{ url_for('verify') }}',
      data: JSON.stringify({
        username: username,
        password: result,
        trajectory: trajectory
      }),
      contentType: 'application/json',
    }).always(function (data, status, xhr) {
      console.log(data, status, xhr);
      if (xhr.status !== 200) {
```

```
            alert('Not allowed');
          } else {
            alert('Ви увійшли в систему як ' + username);
            location.replace('{{ url_for('index') }}');
          }
        });

    });
</script>
```

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО	**ЗАТВЕРДЖУЮ**
Керівник проекту	**В.о. завідувача кафедри**
_____ *О.Г. Жданова*	_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)	(підпис) (ініціали, прізвище)
“16” квітня 2019 р.	“17” квітня 2019 р.

Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання

ТЕХНІЧНЕ ЗАВДАННЯ

Шифр *ДП ІС-4227.1478-с.ТЗ*

на 11 сторінках

Київ – 2019 року

ЗМІСТ

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ 3

1.1 Повне найменування системи та її умовне позначення 3

1.2 Найменування організації-замовника та учасників робіт 3

1.3 Перелік документів, на підставі яких створюється система 3

1.4 Планові терміни початку і закінчення роботи зі створення системи . 4

2 ПРИЗНАЧЕННЯ І МЕТА СТВОРЕННЯ СИСТЕМИ 5

2.1 Призначення системи 5

2.2 Цілі створення системи 5

3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ 6

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 7

4.1 Вимоги до функціональних характеристик 7

4.2 Вимоги до надійності 7

4.3 Вимоги до складу і параметрів технічних засобів 7

5 СТАДІЇ І ЕТАПИ РОЗРОБКИ 10

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ 11

6.1 Види випробувань 11

1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Повне найменування системи та її умовне позначення

Повна назва системи: Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання.

1.2 Найменування організації-замовника та учасників робіт

Генеральним замовником проекту являється кафедра Автоматизованих систем обробки інформації та управління КПІ ім. Сікорського. Представниками замовника є Жданова Олена Григорівна.

Розробником системи є студент групи ІС-42 факультету інформатики та обчислювальної техніки КПІ ім. Ігоря Сікорського Шпаков Віктор Андрійович.

1.3 Перелік документів, на підставі яких створюється система

При розробці системи і створення проектно-експлуатаційної документації Виконавець повинен керуватися вимогами наступних нормативних документів:

- ДСТУ 19.201-78. Технічне завдання. Вимоги до змісту і оформлення;
- ДСТУ 34.601-90. Комплекс стандартів на автоматизовані системи. Автоматизовані системи. Стадії створення;
- ДСТУ 34.201-89. Інформаційні технології. Комплекс стандартів на автоматизовані системи. Види, комплексність і позначення документів при створенні автоматизованих систем.

1.4 Планові терміни початку і закінчення роботи зі створення системи

Плановий термін початку роботи над системою динамічної аутентифікації користувача з використанням лінгвістичного моделювання – 15 лютого 2019 року.

Плановий термін по закінченню роботи над системою динамічної аутентифікації користувача з використанням лінгвістичного моделювання – не пізніше 5 червня 2019 року.

2 ПРИЗНАЧЕННЯ І МЕТА СТВОРЕННЯ СИСТЕМИ

2.1 Призначення системи

Призначенням системи є високоякісне розпізнавання користувача комп’ютерної системи за статистичними характеристиками траєкторії руху курсора миші, вдосконалення персоналізацію користувача для посилення безпеки.

2.2 Цілі створення системи

Метою розробки є покращення захисту даних та забезпечення високої якості розпізнавання користувача комп’ютерної системи або мережі за статистичними характеристиками траєкторії руху курсора миші.

Для досягнення поставленої мети необхідно реалізувати наступні задачі:

- розробити метод оброблення траєкторних даних руху курсора з метою виділення описових статистик та інших ознак, необхідних для розпізнавання користувача;
- розробити модель для розпізнавання користувача у вигляді байєсової мережі (БМ) та методику її застосування до задачі розпізнавання;
- на основі запропонованих методів аналізу даних розробити інформаційну технологію для розпізнавання користувача ЕОМ;
- виконати експериментальні дослідження розробленої інформаційної технології.

3 ХАРАКТЕРИСТИКА ОБ'ЄКТА АВТОМАТИЗАЦІЇ

Для користування сервісом обов'язковою умовою для користувача є наявність браузера, пристроїв вводу/виводу інформації клавіатура, миша або тачпад.

Працювати з сервісом можна користувачам без попередньої реєстрації, проте незареєстровані користувачі не мають доступу до всіх функцій розробки, який мають ті, що зареєструвалися

Об'єктом автоматизації є процес динамічної аутентифікації користувача за допомогою траекторій руху миші.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

Сервіс має створювати умови для реалізації процесу динамічної аутентифікації. Даний сервіс повинен задовольняти потреби користувачів, які цікавляться дослідженням та аналізом різноманітних способів аутентифікації, в тому числі за допомогою біометричних даних.

Сервіс має виконувати наступні функції:

- реєстрація користувача - на основі траекторій вводу за допомогою миші паролю та аналізу характерстик траекторії створити байєсову мережу;
- аутентифікація користувача - дати можливість вже зареєстрованому користувачу пройти аутентифікацію в системі та побачити результат.

4.2 Вимоги до надійності

Програма повинна зберігати працездатність і забезпечувати відновлення своїх функцій при виникненні наступних позаштатних ситуацій:

- при помилках в роботі апаратних засобів (крім носіїв даних і програм).

Програмний продукт повинен поєднувати надійність та функціональність. У разі виникнення аварійних ситуацій необхідно сповіщати користувача та надавати інструкцію для подальших дій.

4.3 Вимоги до складу і параметрів технічних засобів

Характеристики системи, яким повинен відповідати пристрій для коректної роботи системи для клієнта:

Мінімальні системі вимоги

– комп'ютер з:
  1) процесор 1.3 Гц;
  2) оперативна пам'ять 2 Гб;
  3) відеоадаптер VGA (640 x 480);
  4) пристрої вводу/виводу інформації клавітура, миша або тачпад;
– наявність ОС Windows Vista або вище, MacOS 10.10 або вище Ubuntu 12.04 або вище;
– браузер Google Chrome (46.0.3359.181) Firefox (40.0.1) Opera (38.0.2907.68) Яндекс.Браузер (15.3.1) Internet Explorer (8) Safari (3.1.7).

Рекомендовані системі вимоги

– комп'ютер з:
  1) процесор 2.1 Гц;
  2) оперативна пам'ять 4 Гб;
  3) відеоадаптер VGA (1280 x 720);
  4) пристрої вводу/виводу інформації клавіатура, миша або тачпад.
– наявність ОС Windows Vista або вище, MacOS 10.10 або вище, Ubuntu 12.04 або вище;
– браузер Google Chrome (56.0.3359.181) Firefox (46.0.1) Opera (48.0.2907.68) Яндекс.Браузер (17.3.1) Internet Explorer (9) Safari (4.1.7).

Характеристики системи яким повинен відповідати пристрій для коректної роботи системи для серверу:

Комп'ютер з:

– процесор 1.3 Гц;

- оперативна пам’ять 2 Гб;
- відеоадаптер VGA (640 x 480);
- пристрої вводу/виводу інформації клавіатура миша або тачпад;
- наявність ОС Windows 7 або вище, MacOS 10.13 або вище, Ubuntu 14.04 або вище.

5 СТАДІЇ І ЕТАПИ РОЗРОБКИ

Основні етапи виконання робіт з розробки **к**омплексу задач інформаційної підтримки процесу дослідження задач теорії розкладів.

	Назва етапу роботи	**Термін виконання етапу**	**Результат виконання**
.	Підготовка технічного завдання на розробку програмного продукту	17.02.2019	
.	Розробка сценарію роботи	27.02.2019	
.	Технічне проектування – функціональність, модулі, задачі, цілі тощо	03.03.2019	
.	Узгодження з керівником інтерфейсу користувача	07.03.2019	
.	Розробка інформаційного забезпечення	17.03.2019	
.	Розробка програмного забезпечення	29.03.2019	
.	Налагодження програми	13.04.2019	
.	Тестування програми	27.04.2019	
.	Здача готового програмного продукту замовнику	20.05.2019	

6 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ СИСТЕМИ

6.1 Види випробувань

Для контролю коректної роботи програмного забезпечення буде проведено функціональне тестування. В ході тестування буде проведено випробування основних функціональних характеристик системи та цілої системи загалом.

Тестування реєстрації користувача полягає в написанні тестів, які перевіряють можливість бути зареєстрованим в системі зі збереженням вирахованих характеристик у базу даних.

Тестування аутентифікації користувача полягає в написанні тестів, які перевіряють коректність результатів проходження аутентифікації користувача згідно даних його реєстрації.

Тестування інтерфейсу програми полягає в тому, що користувачу надається сторінка з відповідною формою, в яку він вводить тестові дані.

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
Кафедра автоматизованих систем обробки інформації та управління

УЗГОДЖЕНО
Керівник проекту
_____ *О.Г. Жданова*
(підпис) (ініціали, прізвище)
“13” травня 2019 р.

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ *О.А. Павлов*
(підпис) (ініціали, прізвище)
“14” травня 2019 р.

Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання

ПРОГРАМА ТА МЕТОДИКА ВИПРОБУВАНЬ

Шифр *ДП ІС-4227.1478-с.ПМВ*

на 13 сторінках

Київ – 2019 року

ЗМІСТ

1 ОБ'ЄКТ ВИПРОБУВАННЯ 3

1.1 Найменування програми 3

1.2 Область застосування 3

1.3 Умовне позначення програми 3

2 МЕТА ВИПРОБУВАНЬ 4

3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ 5

3.1 Вимоги до функціональних характеристик 5

3.1.1 Вимоги до складу виконуваних функцій 6

4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ 7

5 СКЛАД І ПОРЯДОК ВИПРОБУВАНЬ 8

6 МЕТОДИ ВИПРОБУВАНЬ 9

1 ОБ'ЄКТ ВИПРОБУВАННЯ

1.1 Найменування програми

Найменування системи «Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання».

1.2 Область застосування

Програма використовується для покращення захисту даних шляхом реалізації динамічної аутентифікації, а саме аутентифікації за допомогою траекторій рухів миші користувача.

1.3 Умовне позначення програми

Умовне позначення «DynamicAuthentication».

2 МЕТА ВИПРОБУВАНЬ

Метою випробувань являється перевірка відповідності функцій прикладного програмного забезпечення «Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання».

3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

3.1 Вимоги до функціональних характеристик

Програмний продукт має наступні функціональні характеристики:

- аутентифікація користувача;
- реєстрація користувача.

Функціональна характеристика, як аутентифікація полягає в можливості увійти в систему зареєстрованому користувачу та побачити результати аутентифікації.

Функціональна характеристика, як реєстрація користувача полягає в можливості бути записаним у систему з власним логіном та паролем, з метою подальшої авторизації в системі.

Інтерфейс повинен бути інтуїтивно зрозумілим та зручним. Усі елементи для навігації, як кнопки та посилання повинні бути оформленні без надлишкової кількості графічних елементів та завантажуватися вчасно.

Веб-браузер – середовище взаємодії користувача та прикладного програмного середовища. Взаємодія повина відбуватися за допомогою таких пристроїв, як «мишка» , тачпад та клавіатура. «Мишка» або тачпад повині бути основним інструментом для управління програмним середовищем, а клавіатура повинна забезпечувати заповнення чи редагування необхідних числових або текстових полів. Всі помилки при невірному введені інформації в числові або текстові поля повинні оброблюватися належним чином, а саме повідомленнями з підказками для коректного заповнення. Усі функціональні характеристики повинні бути виконані у повному складі.

3.1.1 Вимоги до складу виконуваних функцій

Тестування виконується на підставі вимог та складу виконуваних функцій, що наведені у технічному завданні.

4 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

Керівництво користувача та код програмного продукту складають програмну документацію.

Основою для випробувань є наступні документи:

– ГОСТ 34.603–92. Інформаційна технологія. Види випробувань автоматизованих систем;

– ГОСТ РД 50-34.698-90. Автоматизовані системи вимог до змісту документів.

5 СКЛАД І ПОРЯДОК ВИПРОБУВАНЬ

Варіанти використання стали основою для тестування. Наведемо обрані сценаріїв:

- сценарій «Реєстрація користувача»;
- сценарій «Відміна реєстрації»;
- сценарій «Перевірка паролю в ході реєстрації»;
- сценарій «Аутентифікація користувача»;
- сценарій «Перевірка паролю під час аутентифікації».

6 МЕТОДИ ВИПРОБУВАНЬ

Функціональне тестування відбувається згідно сценаріїв, що наведені у таблицях 6.1-6.5.

Таблиця 6.1 – Тестовий сценарій «Реєстрація користувача»

Мета тесту:	**Перевірка функції «Реєстрація»**	**Результати тестування**
Початковий стан КЗ	Відкрита головна сторінка	Пройдений
Вхідні данні:	Юзернейм, пароль, траєкторія руху миші.	Пройдений
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Стільки раз набираємо пароль та натискаємо кнопку «Ввести»	Пройдений
Очікуваний результат:	Користувач зареєструвався	Пройдений
Стан АПІМ після проведення випробувань:	Відкрита головна сторінка	Пройдений

Таблиця 6.2 – Тестовий сценарій «Відміна реєстрації»

Мета тесту:	**Перевірка функції «Відміна реєстрації»**	**Результати тестування**
Початковий стан КЗ	Відкрите вікно створення підсистеми	Пройдений
Вхідні данні:	Логін, пароль, траекторія руху миші.	Пройдений
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Після цього натискаємо на кнопку “Почати наново”	Пройдений
Очікуваний результат:	Реєстрація не відбулась	Пройдений
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації	Пройдений

Таблиця 6.3 – Тестовий сценарій «Перевірка паролю в ході реєстрації»

Мета тесту:	**Перевірка функції «Перевірка паролю в ході реєстрації»**	**Результати тестування**
Початковий стан КЗ	Відкрите вікно створення підсистеми	Пройдений
Вхідні данні:	Юзернейм, пароль, траекторія руху миші.	Пройдений
Схема проведення тесту:	Натиснути на посилання “Зареєструватися”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести», після чого отримати повідомлення, скільки разів ще потрібно набрати пароль. Після цього вводимо один раз неправильний пароль та натискаємо «Ввести»	Пройдений
Очікуваний результат:	Реєстрація не відбулась, Отримано повідомлення про помилку.	Пройдений
Стан КЗ після проведення випробувань:	Відкрита сторінка реєстрації, з заповненим логіном та один раз введеним паролем	Пройдений

Таблиця 6.4 – Тестовий сценарій «Аутентифікація користувача»

Мета тесту:	**Перевірка функції «Аутентифікація користувача»**	**Результати тестування**
Початковий стан КЗ	Відкрита головна сторінка	Пройдений
Вхідні данні:	Юзернейм, пароль, траекторія руху миші.	Пройдений
Схема проведення тесту:	Натиснути на посилання “Увійти в систему”, після цього ввести свій логін та набрати пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести»	Пройдений
Очікуваний результат:	Отримане повідомлення, що користувач увійшов в систему	Пройдений
Стан КЗ після проведення випробувань:	Відкрита головна сторінка	Пройдений

Таблиця 6.5 – Тестовий сценарій «Перевірка паролю під час аутентифікації»

Мета тесту:	**Перевірка функції «Перевірка паролю під час аутентифікації»**	**Результати тестування**
Початковий стан КЗ	Відкрита головна сторінка	Пройдений
Вхідні данні:	Юзернейм, пароль, траекторія руху миші.	Пройдений

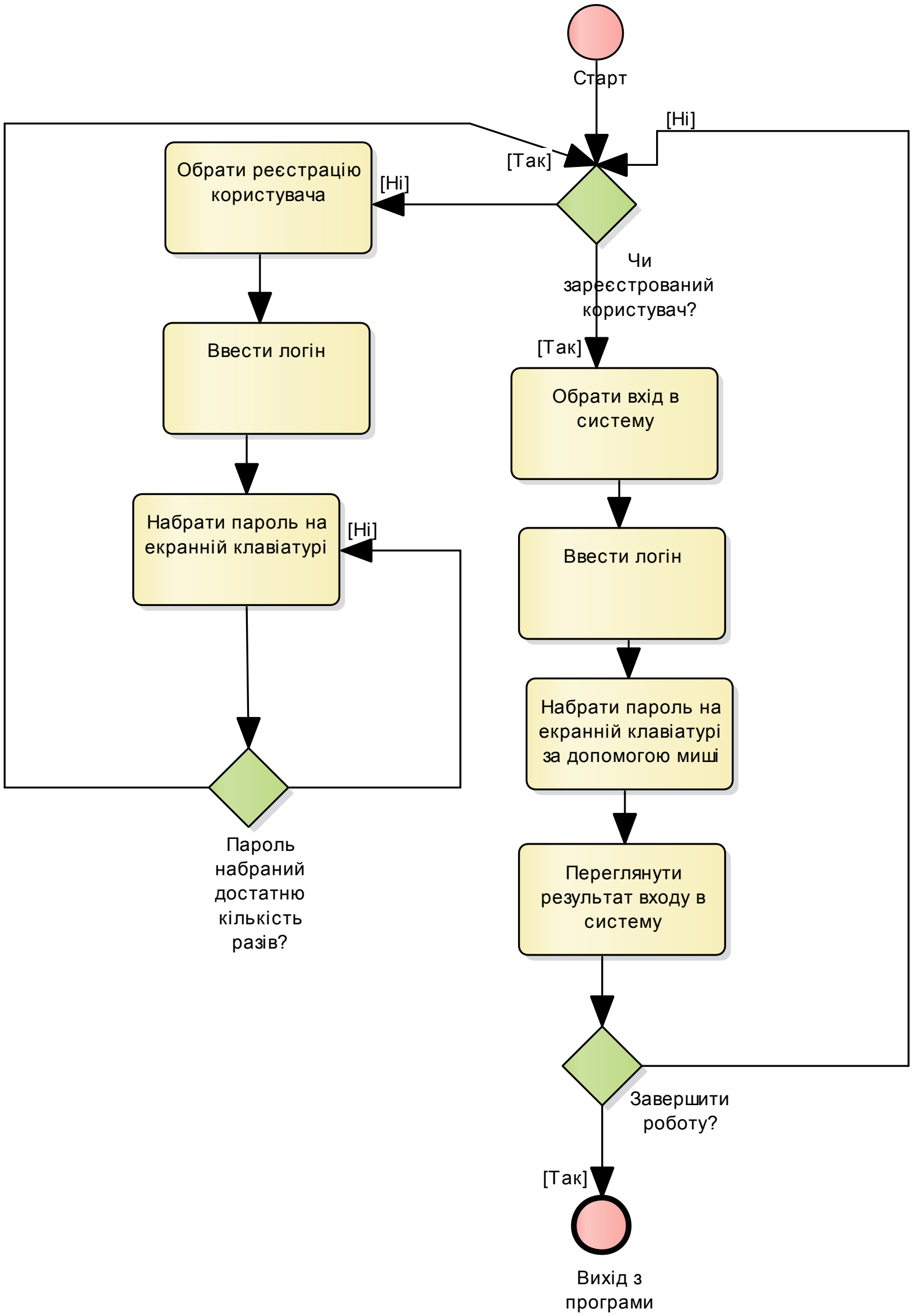
Продовження таблиці 6.5

Мета тесту:	**Перевірка функції «Перевірка паролю під час аутентифікації»**	**Результати тестування**
Схема проведення тесту:	Натиснути на посилання “Увійти в систему”, після цього ввести свій логін та набрати невірний пароль за допомогою миші та екранної клавіатури, далі натиснути на кнопку «Ввести»	Пройдений
Очікуваний результат:	Отримане повідомлення, що доступ не надано, користувач не увійшов в систему	Пройдений
Стан КЗ після проведення випробувань:	Відкрита сторінка входу в систему	Пройдений

Графічний матеріал
до дипломного проекту

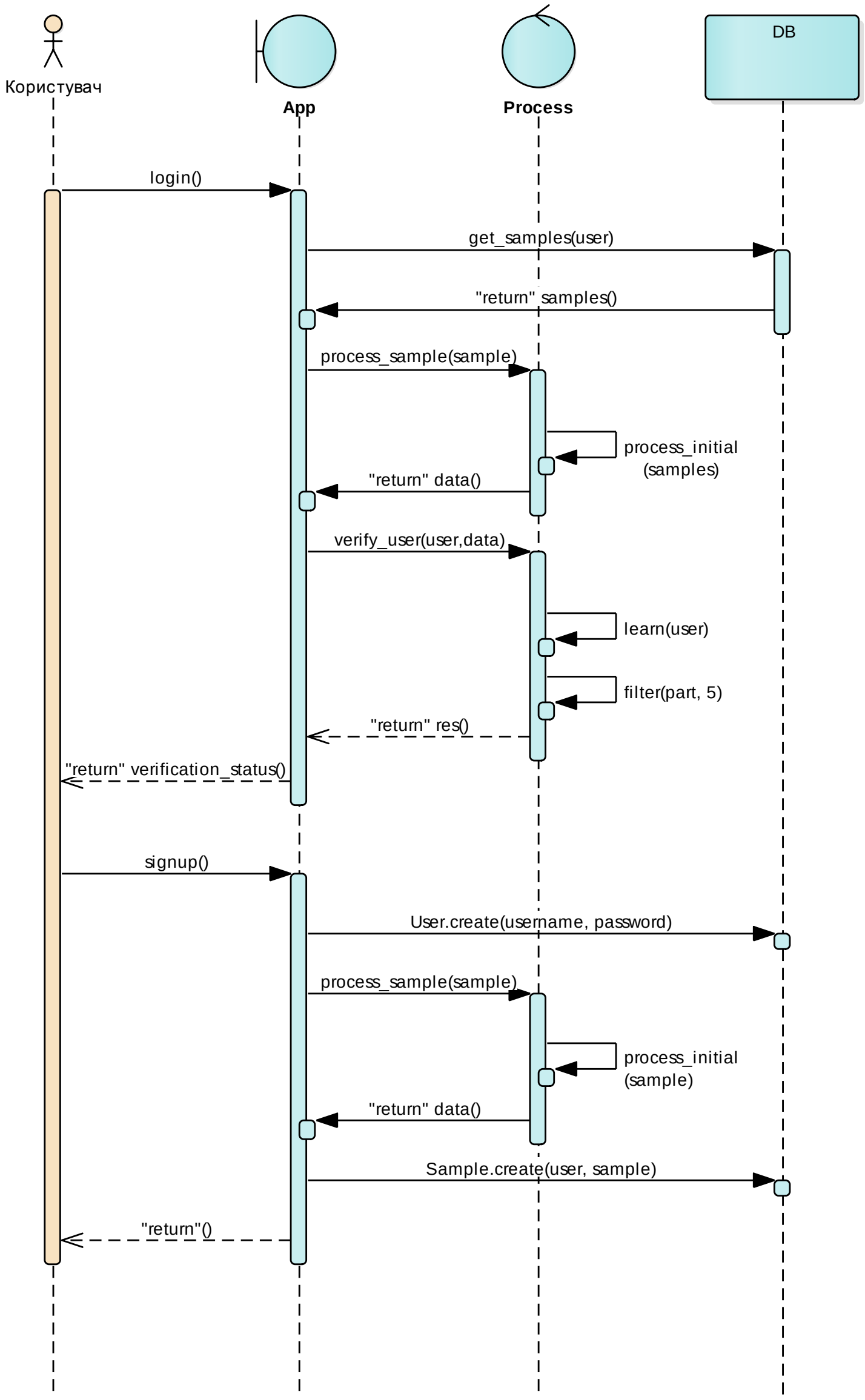
на тему: Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання

Київ – 2019 року



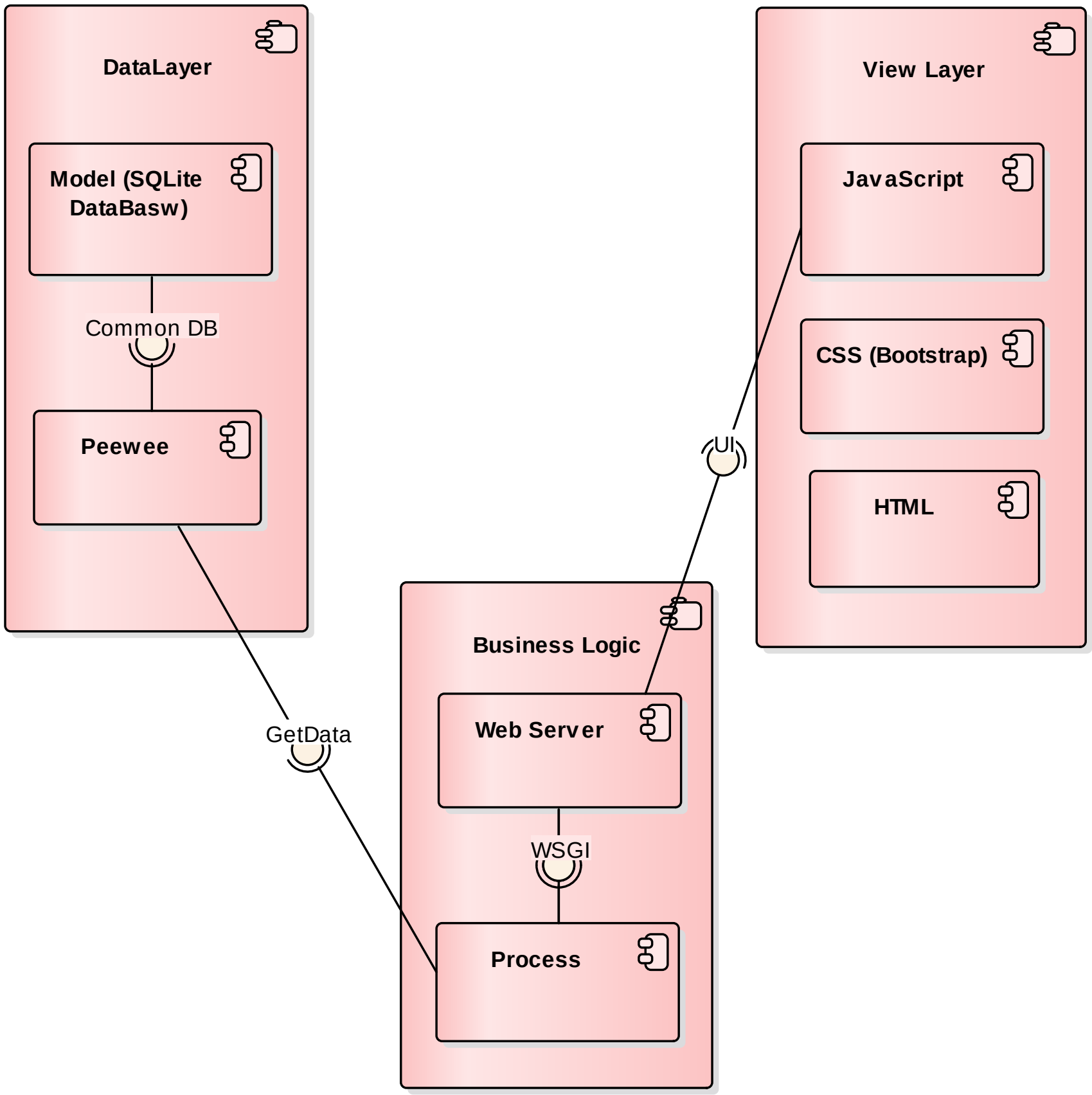
					ДП ІС-4227.1478-с.ССД			
						Лит.	Маса	Масштаб
Зм.	Арк.	№ докум.	Підп.	Дата	Схема структурна діяльності			
Розроб.		Шпаков В.А				Аркуш 1	Аркушів 1	
Перев.		Жданова О.Г.						
Т. Кон.					Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в		
Н. Кон.		Халус О.А.						
Затв.		Жданова О.Г.						

ДП ІС-4227.1478-с.ССП



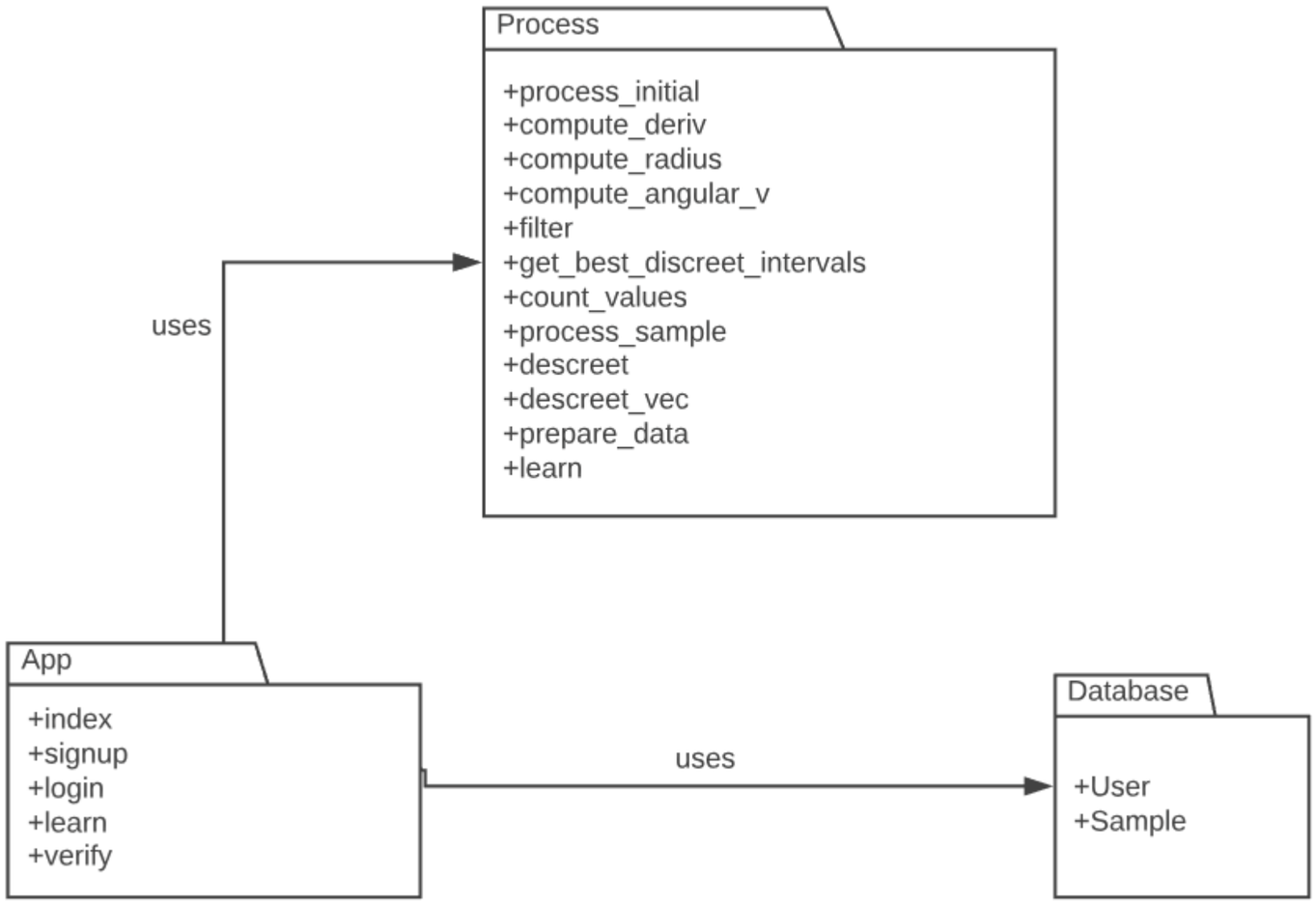
Зм.	Арк.	№ докум.	Підп.	Дата	ДП ІС-4227.1478-с.ССП			
Розроб.		Шпаков В.А.			Схема структурна послідовності	Лит.	Маса	Масштаб
Перев.		Жданова О.Г.				Аркуш 1	Аркушів 1	
Т. Кон.					Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в		
Н. Кон.		Халус О.А.						
Затв.		Жданова О.Г.						

ДП ІС-4227.1478-с.ССК



					ДП ІС-4227.1478-с.ССК	
					Схема структурна компонентів	Лит. / Маса / Масштаб
Зм.	Арк.	№ докум.	Підп.	Дата		Аркуш 1 / Аркушів 1
Розроб.		Шпаков В.А.			Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в
Перев.		Жданова О.Г.				
Т. Кон.						
Н. Кон.		Халус О.А.				
Затв.		Жданова О.Г.				

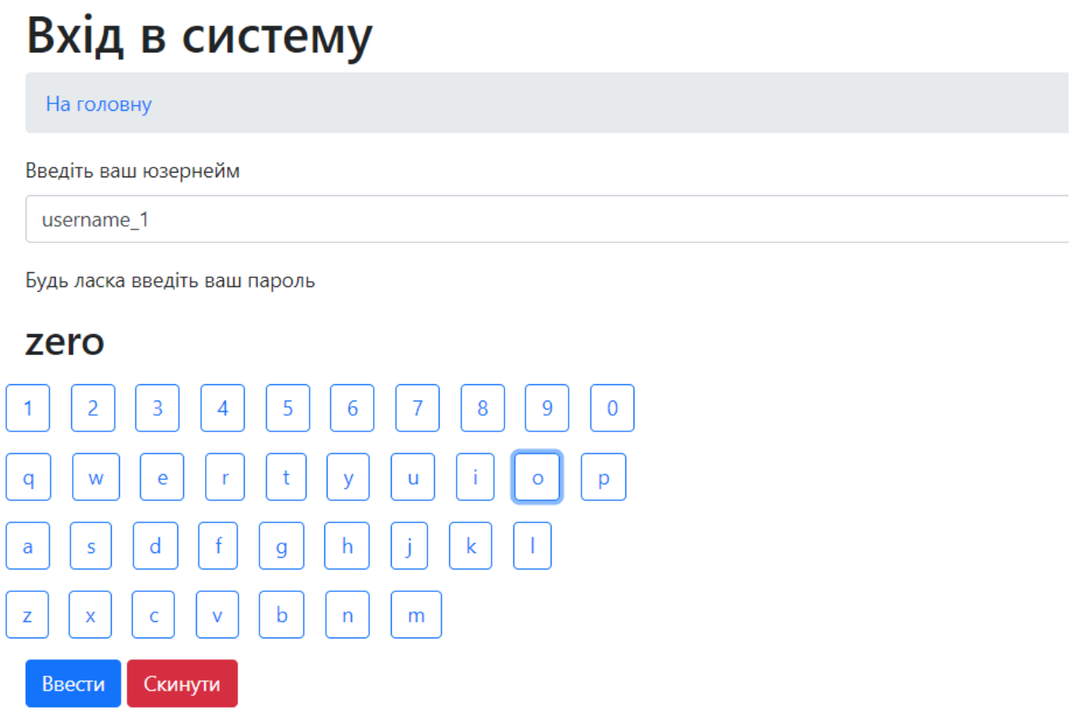
ДП ІС-4227.1478-с.ССП



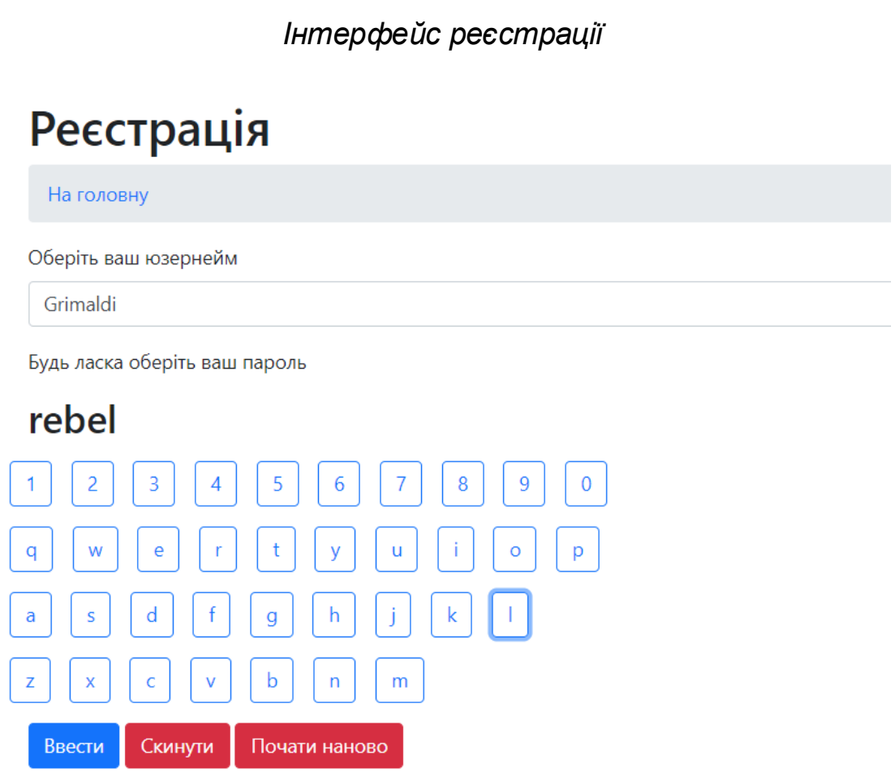
					ДП ІС-4227.1478-с.ССП		
						Літера	Маса	Масштаб
Зм.	Арк.	№ документа	Підпис	Дата	Схема структурна пакетів			
Розробив		Шпаков В.А.				Аркуш 1	Аркушів 1
Перевірив		Жданова О.Г.					
Т. кон.							
					Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в	
Н. кон.		Халус О.А.					
Затвердив		Жданова О.Г.					

					ДП ІС-4227.1478-с.СБД		
Зм.	Арк.	№ документа	Підпис	Дата	Схема бази даних	Літера	Маса / Масштаб
Розробив		Шпаков В.А.					
Перевірив		Жданова О.Г.				Аркуш 1	Аркушів 1
Т. кон.					Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в	
Н. кон.		Халус О.А.					
Затвердив		Жданова О.Г.					

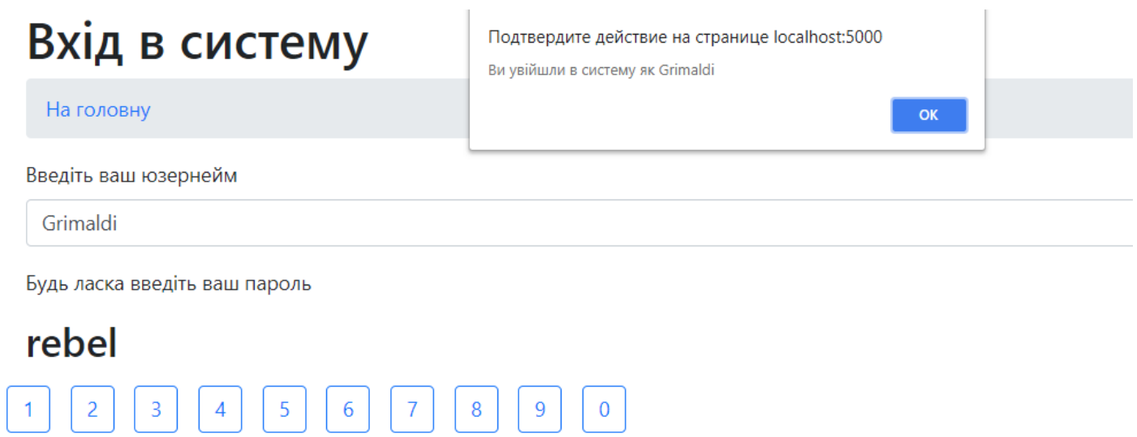
Інтерфейс для авторизації



Інтерфейс реєстрації



Інтерфейс перегляду результату авторизації



					ДП ІС-4227.1478-с.КЕ		
					Креслення вигляду екранних форм	Літера	Маса / Масштаб
Зм.	Арк.	№ документа	Підпис	Дата			
Розроб.		Шпаков В.А				Аркуш 1	Аркушів 1
Перевірив		Жданова О.Г.			Система динамічної аутентифікації користувача з використанням лінгвістичного моделювання	КПІ ім. Ігоря Сікорського кафедра АСОІУ гр. ІС-51в	
Т. кон.							
Н. кон.		Халус О.А.					
Затвердив		Жданова О.Г.					